

LPIC-1

Versión 5.0
Español



Table of Contents



Tema 101: Arquitectura del Sistema



101.1 Determinar y configurar los ajustes de hardware

Referencia al objetivo del LPI

LPIC-1 v5, Exam 101, Objective 101.1

Importancia

2

Áreas de conocimiento clave

- Activar y desactivar los periféricos integrados.
- Diferenciar entre los distintos tipos de dispositivos de almacenamiento masivo.
- Determinar los recursos de hardware para los dispositivos.
- Herramientas y utilidades para listar información de hardware (por ejemplo, lsusb, lspci, etc.).
- Herramientas y utilidades para manipular dispositivos USB.
- Conocimientos conceptuales de sysfs, udev y dbus.

Lista parcial de archivos, términos y utilidades

- `/sys/`
- `/proc/`
- `/dev/`
- `modprobe`
- `lsmod`
- `lspci`
- `lsusb`



101.1 Lección 1

Certificación:	LPIC-1
Versión:	5.0
Tema:	101 Arquitectura del Sistema
Objetivo:	101.1 Determinar y configurar hardware
Lección:	1 de 1

Introducción

Desde los inicios de la informática electrónica, los fabricantes de computadoras personales y de negocios han integrado una variedad de partes de hardware en sus máquinas, que a su vez deben ser compatibles con el sistema operativo. Eso podría ser abrumador desde la perspectiva del desarrollador del sistema operativo, a menos que la industria establezca estándares para los conjuntos de instrucciones y la comunicación del dispositivo. Al igual que la capa de abstracción estandarizada proporcionada por el sistema operativo a una aplicación, estos estándares facilitan la escritura y el mantenimiento de un sistema operativo que no está vinculado a un modelo de hardware específico. Sin embargo, la complejidad del hardware subyacente integrado a veces requiere ajustes sobre cómo deben exponerse los recursos al sistema operativo, para que pueda instalarse y funcionar correctamente.

Algunos de estos ajustes pueden realizarse incluso sin un sistema operativo instalado. La mayoría de las máquinas ofrecen una utilidad de configuración que se puede ejecutar cuando se enciende la máquina. Hasta mediados de los 2000, la utilidad de configuración se implementó en el BIOS (*Basic Input/Output System*), el estándar para el firmware que contiene las rutinas de configuración básicas que se encuentran en las placas base x86. Desde finales de la primera década de los 2000, las máquinas basadas en la arquitectura x86 comenzaron a reemplazar el BIOS con una nueva

implementación llamada UEFI (*Unified Extensible Firmware Interface*), que tiene características más sofisticadas para la identificación, prueba, configuración y actualizaciones de firmware. A pesar del cambio, no es raro llamar a la utilidad de configuración BIOS, ya que ambas implementaciones cumplen el mismo propósito básico.

NOTE En la siguiente lección se cubrirán más detalles sobre las similitudes y diferencias entre BIOS y UEFI.

Activación de Dispositivos

La utilidad de configuración del sistema se presenta después de presionar una tecla específica cuando se enciende la computadora. La tecla que debe presionar varía de un fabricante a otro, pero generalmente es `Del` o una de las teclas de función, como `F2` o `F12`. Generalmente, la combinación de teclas para iniciar la configuración del BIOS se muestra en la pantalla al iniciar la máquina

En la utilidad de configuración del BIOS, es posible habilitar y deshabilitar periféricos integrados, activar la protección básica contra errores y cambiar configuraciones de hardware como IRQ (solicitud de interrupción) y DMA (acceso directo a memoria). Raramente se necesita cambiar esta configuración en las máquinas modernas, pero puede ser necesario hacer ajustes para abordar problemas específicos. Existen tecnologías RAM, por ejemplo, que son compatibles con velocidades de transferencia de datos más rápidas que los valores predeterminados, por lo que se recomienda cambiarlo a los valores especificados por el fabricante. Algunas CPU ofrecen características que pueden no ser necesarias para una instalación en particular y pueden desactivarse. Las funciones deshabilitadas reducirán el consumo de energía y pueden aumentar la protección del sistema, ya que las funciones de la CPU que contienen errores conocidos también se pueden deshabilitar.

Si la máquina está equipada con muchos dispositivos de almacenamiento, es importante definir cuál tiene el gestor de arranque correcto y debe ser la primera entrada en el orden de arranque del dispositivo. Es posible que el sistema operativo no se cargue si el dispositivo incorrecto aparece primero en la lista de verificación de arranque del BIOS.

Inspección de dispositivos en Linux

Una vez que los dispositivos se identifican correctamente, corresponde al sistema operativo asociar los componentes de software correspondientes requeridos por ellos. Cuando una característica de hardware no funciona como se esperaba, es importante identificar dónde está sucediendo exactamente el problema. Cuando el sistema operativo no detecta un dispositivo, lo más probable es que éste, o el puerto al que está conectado, esté defectuoso. Cuando el dispositivo se detecta correctamente, pero aún no funciona como se espera, puede haber un problema en el lado del sistema operativo. Por lo tanto, uno de los primeros pasos cuando se trata con problemas relacionados con el hardware es verificar si el sistema operativo está detectando correctamente el

dispositivo. Hay dos formas básicas de identificar recursos de hardware en un sistema Linux: usar comandos especializados o leer archivos específicos dentro de sistemas de archivos especiales.

Comandos para inspección

Dos comandos esenciales para identificar dispositivos conectados en Linux son:

`lspci`
: Muestra todos los dispositivos actualmente conectados al bus PCI (*Peripheral Component Interconnect*). Los dispositivos PCI pueden ser un componente conectado a la placa base, como un controlador de disco, o una tarjeta de expansión instalada en una ranura PCI, como una tarjeta gráfica externa.

`lsusb`
: Enumera los dispositivos USB (*Universal Serial Bus*) actualmente conectados a la máquina. Aunque existen dispositivos USB para casi cualquier propósito imaginable, la interfaz USB se utiliza en gran medida para conectar dispositivos de entrada (teclados, dispositivos señaladores) y medios de almacenamiento extraíbles.

La salida de los comandos `lspci` y `lsusb` consiste en una lista de todos los dispositivos PCI y USB identificados por el sistema operativo. Sin embargo, es posible que el dispositivo aún no esté completamente operativo, porque cada parte del hardware requiere de un componente de software para controlar el dispositivo correspondiente. Este componente de software se denomina *módulo del kernel* y puede formar parte del núcleo oficial de Linux o agregarse por separado de otras fuentes.

Los módulos del núcleo de Linux relacionados con dispositivos de hardware también se denominan controladores, como en otros sistemas operativos. Sin embargo, los controladores para Linux no siempre son suministrados por el fabricante del dispositivo. Mientras que algunos fabricantes proporcionan sus propios controladores binarios para que se instalen por separado, muchos controladores están escritos por desarrolladores independientes. Históricamente, las partes que funcionan en Windows, por ejemplo, pueden no tener un módulo del núcleo equivalente para Linux. Hoy en día, los sistemas operativos basados en Linux tienen un fuerte soporte de hardware y la mayoría de los dispositivos funcionan sin esfuerzo.

Los comandos directamente relacionados con el hardware a menudo requieren privilegios de root para ejecutarse o solo mostrarán información limitada cuando los ejecute un usuario normal, por lo que puede ser necesario iniciar sesión como root o ejecutar el comando con `sudo`.

La siguiente salida del comando `lspci`, por ejemplo, muestra algunos dispositivos identificados:

```
$ lspci
01:00.0 VGA compatible controller: NVIDIA Corporation GM107 [GeForce GTX 750 Ti]
(rev a2)
04:02.0 Network controller: Ralink corp. RT2561/RT61 802.11g PCI
04:04.0 Multimedia audio controller: VIA Technologies Inc. ICE1712 [Envy24] PCI
Multi-Channel I/O Controller (rev 02)
04:0b.0 FireWire (IEEE 1394): LSI Corporation FW322/323 [TrueFire] 1394a Controller
(rev 70)
```

La salida de dichos comandos pueden tener decenas de líneas, por lo que los ejemplos contienen solo las secciones de interés. Los números hexadecimales al principio de cada línea son las direcciones únicas del dispositivo PCI correspondiente. El comando `lspci` muestra más detalles sobre un dispositivo específico si su dirección se da con la opción `-s`, acompañada de la opción `-v`:

```
$ lspci -s 04:02.0 -v
04:02.0 Network controller: Ralink corp. RT2561/RT61 802.11g PCI
    Subsystem: Linksys WMP54G v4.1
    Flags: bus master, slow devsel, latency 32, IRQ 21
    Memory at e3100000 (32-bit, non-prefetchable) [size=32K]
    Capabilities: [40] Power Management version 2
    kernel driver in use: rt61pci
```

El resultado ahora muestra muchos más detalles del dispositivo en la dirección `04:02.0`. Es un controlador de red, cuyo nombre interno es `Ralink corp. RT2561/RT61 802.11g PCI`. `Subsystem` está asociado con la marca y el modelo del dispositivo (`Linksys WMP54G v4.1`) y puede ser útil para fines de diagnóstico.

El módulo del núcleo del sistema operativo se puede identificar en la línea `kernel driver in use`, que muestra el módulo `rt61pci`. De toda la información recopilada, es correcto suponer que:

1. El dispositivo ha sido identificado.
2. Se cargó un módulo en el núcleo del sistema operativo.
3. El dispositivo debe estar listo para usarse.

La opción `-k`, disponible en versiones más recientes de `lspci`, proporciona otra forma de verificar qué módulo del núcleo del sistema operativo está en uso para el dispositivo especificado:

```
$ lspci -s 01:00.0 -k
01:00.0 VGA compatible controller: NVIDIA Corporation GM107 [GeForce GTX 750 Ti]
(rev a2)
    kernel driver in use: nvidia
    kernel modules: nouveau, nvidia_drm, nvidia
```

Para el dispositivo elegido, una placa GPU NVIDIA, `lspci` indica que el módulo en uso se llama `nvidia`, en la línea `kernel driver in use: nvidia` y todos los módulos del núcleo del sistema operativo correspondientes se enumeran en la línea `kernel modules: nouveau , nvidia_drm, nvidia`.

El comando `lsusb` es similar a `lspci`, pero enumera la información de USB exclusivamente:

```
$ lsusb
Bus 001 Device 029: ID 1781:0c9f Multiple Vendors USBtiny
Bus 001 Device 028: ID 093a:2521 Pixart Imaging, Inc. Optical Mouse
Bus 001 Device 020: ID 1131:1001 Integrated System Solution Corp. KY-BT100
Bluetooth Adapter
Bus 001 Device 011: ID 04f2:0402 Chicony Electronics Co., Ltd Genius LuxeMate i200
Keyboard
Bus 001 Device 007: ID 0424:7800 Standard Microsystems Corp.
Bus 001 Device 003: ID 0424:2514 Standard Microsystems Corp. USB 2.0 Hub
Bus 001 Device 002: ID 0424:2514 Standard Microsystems Corp. USB 2.0 Hub
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
```

El comando `lsusb` muestra los canales USB disponibles y los dispositivos conectados a ellos. Al igual que con `lspci`, la opción `-v` muestra una salida más detallada. Se puede seleccionar un dispositivo específico para inspección proporcionando su ID a la opción `-d`:

```
$ lsusb -v -d 1781:0c9f
Bus 001 Device 029: ID 1781:0c9f Multiple Vendors USBtiny
Device Descriptor:
  bLength                18
  bDescriptorType         1
  bcdUSB               1.01
  bDeviceClass          255 Vendor Specific Class
  bDeviceSubClass         0
  bDeviceProtocol         0
  bMaxPacketSize0         8
  idVendor           0x1781 Multiple Vendors
  idProduct          0x0c9f USBtiny
  bcdDevice            1.04
  iManufacturer           0
  iProduct                2 USBtiny
  iSerial                 0
  bNumConfigurations      1
```

Con la opción `-t`, el comando `lsusb` muestra las asignaciones actuales de los dispositivos USB en forma de árbol jerárquico:

```
$ lsusb -t
/:  Bus 01.Port 1: Dev 1, Class=root_hub, Driver=dwc_otg/1p, 480M
    |__ Port 1: Dev 2, If 0, Class=Hub, Driver=hub/4p, 480M
        |__ Port 1: Dev 3, If 0, Class=Hub, Driver=hub/3p, 480M
            |__ Port 2: Dev 11, If 1, Class=Human Interface Device, Driver=usbhid,
1.5M
            |__ Port 2: Dev 11, If 0, Class=Human Interface Device, Driver=usbhid,
1.5M
            |__ Port 3: Dev 20, If 0, Class=Wireless, Driver=btusb, 12M
            |__ Port 3: Dev 20, If 1, Class=Wireless, Driver=btusb, 12M
            |__ Port 3: Dev 20, If 2, Class=Application Specific Interface,
Driver=, 12M
            |__ Port 1: Dev 7, If 0, Class=Vendor Specific Class, Driver=lan78xx,
480M
        |__ Port 2: Dev 28, If 0, Class=Human Interface Device, Driver=usbhid, 1.5M
        |__ Port 3: Dev 29, If 0, Class=Vendor Specific Class, Driver=, 1.5M
```

Es posible que no todos los dispositivos tengan los módulos correspondientes asociados. La comunicación con determinados dispositivos puede ser manejada directamente por la aplicación, sin la intermediación que proporciona un módulo. Sin embargo, hay información significativa en la salida proporcionada por `lsusb -t`. Cuando existe un módulo coincidente, su nombre aparece al final

de la línea del dispositivo, como en `Driver=btusb`. El dispositivo `Class` identifica la categoría general, como `Human Interface Device`, `Wireless`, `Vendor Specific Class`, `Mass Storage`, entre otros. Para verificar qué dispositivo está usando el módulo `btusb`, presente en la lista anterior, se deben dar tanto los números de `Bus` como de `Dev` a la opción `-s` del comando `lsusb`:

```
$ lsusb -s 01:20
Bus 001 Device 020: ID 1131:1001 Integrated System Solution Corp. KY-BT100
Bluetooth Adapter
```

Es común tener un gran conjunto de módulos del núcleo del sistema operativo cargados en un sistema Linux estándar en cualquier momento. La forma preferible de interactuar con ellos es usar los comandos proporcionados por el paquete `kmod`, que es un conjunto de herramientas para manejar tareas comunes con módulos del kernel de Linux como insertar, eliminar, enumerar, verificar propiedades, resolver dependencias y alias. El comando `lsmod`, por ejemplo, muestra todos los módulos cargados actualmente:

```
$ lsmod
Module                  Size  Used by
kvm_intel             138528  0
kvm                   421021  1 kvm_intel
iTCO_wdt               13480  0
iTCO_vendor_support    13419  1 iTCO_wdt
snd_usb_audio         149112  2
snd_hda_codec_realtek  51465  1
snd_ice1712            75006  3
snd_hda_intel          44075  7
arc4                   12608  2
snd_cs8427             13978  1 snd_ice1712
snd_i2c                13828  2 snd_ice1712,snd_cs8427
snd_ice17xx_ak4xxx     13128  1 snd_ice1712
snd_ak4xxx_adda        18487  2 snd_ice1712,snd_ice17xx_ak4xxx
microcode              23527  0
snd_usbmidi_lib        24845  1 snd_usb_audio
gspca_pac7302          17481  0
gspca_main             36226  1 gspca_pac7302
videodev              132348  2 gspca_main,gspca_pac7302
rt61pci                32326  0
rt2x00pci              13083  1 rt61pci
media                  20840  1 videodev
rt2x00mmio             13322  1 rt61pci
hid_dr                 12776  0
snd_mpu401_uart        13992  1 snd_ice1712
rt2x00lib              67108  3 rt61pci,rt2x00pci,rt2x00mmio
snd_rawmidi            29394  2 snd_usbmidi_lib,snd_mpu401_uart
```

La salida del comando `lsmod` se divide en tres columnas:

`Module`

Nombre del módulo.

`Size`

Cantidad de memoria RAM ocupada por el módulo, en bytes.

`Used by`

Módulos dependientes.

Algunos módulos requieren que otros módulos funcionen correctamente, como es el caso de los módulos para dispositivos de audio:

```
$ lsmod | fgrep -i snd_hda_intel
snd_hda_intel          42658  5
snd_hda_codec         155748  3 snd_hda_codec_hdmi,snd_hda_codec_via,snd_hda_intel
snd_pcm                81999  3 snd_hda_codec_hdmi,snd_hda_codec,snd_hda_intel
snd_page_alloc         13852  2 snd_pcm,snd_hda_intel
snd                    59132  19
snd_hwdep,snd_timer,snd_hda_codec_hdmi,snd_hda_codec_via,snd_pcm,snd_seq,snd_hda_co
dec,snd_hda_intel,snd_seq_device
```

La tercera columna, `Used by`, muestra los módulos que requieren que el módulo de la primera columna funcione correctamente. Muchos módulos de la arquitectura de sonido de Linux, con el prefijo `snd`, son interdependientes. Al buscar problemas durante el diagnóstico del sistema, puede ser útil descargar módulos específicos actualmente cargados. El comando `modprobe` se puede usar tanto para cargar como para descargar módulos del núcleo del sistema operativo: para descargar un módulo y sus módulos relacionados, siempre que no estén siendo utilizados por un proceso en ejecución, se debe usar el comando `modprobe -r`. Por ejemplo, para descargar el módulo `snd-hda-intel` (el módulo para un dispositivo de audio Intel HDA) y otros módulos relacionados con el sistema de sonido:

```
# modprobe -r snd-hda-intel
```

Además de cargar y descargar módulos del núcleo del sistema operativo mientras el sistema se está ejecutando, es posible cambiar los parámetros del módulo cuando se está cargando el núcleo del sistema operativo, no muy diferente de pasar opciones a comandos. Cada módulo acepta parámetros específicos, pero la mayoría de las veces se recomiendan los valores predeterminados y no se necesitan parámetros adicionales. Sin embargo, en algunos casos es necesario usar parámetros para cambiar el comportamiento de un módulo para que funcione como se espera.

Usando el nombre del módulo como único argumento, el comando `modinfo` muestra una descripción, el archivo, el autor, la licencia, la identificación, las dependencias y los parámetros disponibles para el módulo dado. Los parámetros personalizados para un módulo pueden hacerse persistentes al incluirlos en el archivo `/etc/modprobe.conf` o en archivos individuales con la extensión `.conf` en el directorio `/etc/modprobe.d/`. La opción `-p` hará que el comando `modinfo` muestre todos los parámetros disponibles e ignore la otra información:

```
# modinfo -p nouveau
vram_pushbuf:Create DMA push buffers in VRAM (int)
tv_norm:Default TV norm.
                Supported: PAL, PAL-M, PAL-N, PAL-Nc, NTSC-M, NTSC-J,
                        hd480i, hd480p, hd576i, hd576p, hd720p, hd1080i.
                Default: PAL
                NOTE Ignored for cards with external TV encoders. (charp)
nofbaccel:Disable fbcon acceleration (int)
fbcon_bpp:fbcon bits-per-pixel (default: auto) (int)
mst:Enable DisplayPort multi-stream (default: enabled) (int)
tv_disable:Disable TV-out detection (int)
ignorelid:Ignore ACPI lid status (int)
duallink:Allow dual-link TMDS (default: enabled) (int)
hdmimhz:Force a maximum HDMI pixel clock (in MHz) (int)
config:option string to pass to driver core (charp)
debug:debug string to pass to driver core (charp)
noaccel:disable kernel/abi16 acceleration (int)
modeset:enable driver (default: auto, 0 = disabled, 1 = enabled, 2 = headless)
(int)
atomic:Expose atomic ioctl (default: disabled) (int)
runpm:disable (0), force enable (1), optimus only default (-1) (int)
```

La salida de muestra todos los parámetros disponibles para el módulo `nouveau`, un módulo de kernel proporcionado por *Nouveau Project* como alternativa a los controladores propietarios para tarjetas GPU NVIDIA. La opción `modeset`, por ejemplo, permite controlar si la resolución y la profundidad de la pantalla se establecerán en el espacio del kernel en lugar del espacio del usuario. Agregar `options nouveau modeset=0` al archivo `/etc/modprobe.d/nouveau.conf` deshabilitará la función del kernel de modeset.

Si un módulo está causando problemas, el archivo `/etc/modprobe.d/blacklist.conf` puede usarse para bloquear la carga del módulo. Por ejemplo, para evitar la carga automática del módulo `nouveau`, la línea `blacklist nouveau` debe agregarse al archivo `/etc/modprobe.d/blacklist.conf`. Esta acción es necesaria cuando el módulo propietario `nvidia` está instalado y el módulo predeterminado `nouveau` debe ignorarse.

NOTE

Puede modificar el archivo `/etc/modprobe.d/blacklist.conf` que ya existe en el sistema de forma predeterminada. Sin embargo, el método preferido es crear un archivo de configuración separado, `/etc/modprobe.d/<module_name>.conf`, que contendrá configuraciones específicas solo para el módulo del núcleo dado.

Archivos de información y archivos de dispositivo

Los comandos `lspci`,`lsusb` y `lsmod` actúan como interfaz para leer la información del dispositivo almacenada por el sistema operativo. Este tipo de información se guarda en archivos especiales en los directorios `/proc` y `/sys`. Estos directorios son puntos de montaje para sistemas de archivos que no están presentes en una partición de dispositivo, sino solo en el espacio RAM utilizado por el núcleo del sistema operativo para almacenar la configuración en tiempo de ejecución y la información sobre los procesos en ejecución. Dichos sistemas de archivos no están destinados al almacenamiento convencional de archivos, por lo que se denominan pseudo-sistemas de archivos y solo existen mientras el sistema se está ejecutando. El directorio `/proc` contiene archivos con información sobre procesos en ejecución y recursos de hardware. Algunos de los archivos importantes en `/proc` para inspeccionar el hardware son:

`/proc/cpuinfo`
: Enumera información detallada sobre las CPU encontradas por el sistema operativo.

`/proc/interrupts`
: Una lista de números de las interrupciones por dispositivo de entrada/salida para cada CPU.

`/proc/ioports`
: Enumera los puertos de entrada/salida registrados actualmente en uso.

`/proc/dma`
: Enumera los canales DMA (acceso directo a memoria) registrados en uso.

Los archivos dentro del directorio `/sys` tienen roles similares a los de `/proc`. Sin embargo, el directorio `/sys` tiene el propósito específico de almacenar información del dispositivo y datos del kernel relacionados con el hardware, mientras que `/proc` también contiene información sobre varias estructuras de datos del kernel, incluidos los procesos en ejecución y la configuración.

Otro directorio directamente relacionado con dispositivos en un sistema Linux estándar es `/dev`. Cada archivo dentro de `/dev` está asociado con un dispositivo del sistema, particularmente dispositivos de almacenamiento. Un disco duro IDE heredado, por ejemplo, cuando está conectado al primer canal IDE de la placa base, está representado por el archivo `/dev/hda`. Cada partición en este disco será identificada por `/dev/hda1`,`/dev/hda2` hasta la última partición encontrada.

Los dispositivos extraíbles son manejados por el subsistema *udev*, que crea los dispositivos correspondientes en `/dev`. El núcleo de Linux captura el evento de detección de hardware y lo pasa al proceso udev, que luego identifica el dispositivo y crea dinámicamente los archivos correspondientes en `/dev`, utilizando reglas predefinidas.

En las distribuciones actuales de Linux, udev es responsable de la identificación y configuración de los dispositivos que ya están presentes durante el encendido de la máquina (*coldplug detection*) y los dispositivos identificados mientras el sistema está en funcionamiento (*hotplug detection*). Udev se basa en *SysFS*, el pseudo sistema de archivos para la información relacionada con los dispositivos montados en `/sys`.

> **NOTE**
> Hotplug es el término utilizado para referirse a la detección y configuración de un dispositivo mientras el sistema está en funcionamiento, como cuando se inserta un dispositivo USB. El núcleo de Linux ha admitido funciones de conexión en caliente desde la versión 2.6, lo que permite que la mayoría de los buses del sistema (PCI, USB, etc.) activen eventos de conexión en caliente cuando un dispositivo está conectado o desconectado.

A medida que se detectan nuevos dispositivos, udev busca una regla coincidente en las reglas predefinidas almacenadas en el directorio `/etc/udev/rules.d/`. La distribución proporciona las reglas más importantes, pero se pueden agregar nuevas para casos específicos.

Dispositivos de Almacenamiento

En Linux, los dispositivos de almacenamiento se denominan genéricamente dispositivos de bloque, porque los datos se leen desde y hacia estos dispositivos en bloques de datos almacenados en búfer con diferentes tamaños y posiciones. Cada dispositivo de bloque se identifica mediante un archivo en el directorio `/dev`, con el nombre del archivo según el tipo de dispositivo (IDE, SATA, SCSI, etc.) y sus particiones. Los dispositivos de CD/DVD y disquetes, por ejemplo, recibirán sus nombres correspondientes en `/dev`: una unidad de CD/DVD conectada al segundo canal IDE se identificará como `/dev/hdc` (`/dev/hda` y `/dev/hdb` están reservados para los dispositivos maestro y esclavo en el primer canal IDE) y una unidad de disquete antigua se identificará como `/dev/fd0`, `/dev/fd1`, etc.

Desde la versión 2.4 del kernel de Linux en adelante, la mayoría de los dispositivos de almacenamiento ahora se identifican como si fueran dispositivos SCSI, independientemente de su tipo de hardware. Los dispositivos de bloque IDE, SSD y USB tendrán el prefijo `sd`. Para los discos IDE, se utilizará el prefijo `sd`, pero se elegirá la tercera letra dependiendo de si la unidad es maestra o esclava (en el primer canal IDE, el maestro será `sda` y el esclavo será `sdb`) Las particiones se listan numéricamente. Las rutas `/dev/sda1`, `/dev/sda2`, etc. se usan para la primera y segunda partición del dispositivo de bloque identificado primero y `/dev/sdb1`, `/dev/sdb2`, etc. identifique las particiones primera y segunda del dispositivo de bloque identificado en segundo lugar. La excepción a este patrón ocurre con las tarjetas de memoria (tarjetas SD) y los dispositivos NVMe (SSD conectados al bus PCI Express). Para las tarjetas SD, las rutas `/dev/mmcblk0p1`, `/dev/mmcblk0p2`, etc. se utilizan para la primera y segunda partición del dispositivo identificado primero y `/dev/mmcblk1p1`, `/dev/mmcblk1p2`, etc. utilizado para identificar la primera y la segunda partición del dispositivo identificado en segundo lugar. Los dispositivos NVMe reciben el prefijo `nvme`, como en

`/dev/nvme0n1p1` y `/dev/nvme0n1p2`.

Ejercicios Guiados

1. Supongamos que un sistema operativo no puede iniciarse después de agregar un segundo disco SATA al sistema. Sabiendo que ninguno de los dispositivos está defectuoso, ¿cuál podría ser la posible causa de este error?

2. Suponga que desea asegurarse de que la tarjeta de video externa conectada al bus PCI de su computadora de escritorio recién adquirida realmente sea la anunciada por el fabricante, pero al abrir el cajón de la computadora anularía la garantía. ¿Qué comando podría usarse para enumerar los detalles de la tarjeta de video, tal como fueron detectados por el sistema operativo?

3. La siguiente línea es parte de la salida generada por el comando `lspci`:

   ```
   03:00.0 RAID bus controller: LSI Logic / Symbios Logic MegaRAID SAS 2208
   [Thunderbolt] (rev 05)
   ```

   ¿Qué comando debe ejecutar para identificar el módulo del núcleo del sistema operativo en uso para este dispositivo específico?

4. Un administrador del sistema quiere probar diferentes parámetros para el módulo del kernel `bluetooth` sin reiniciar el sistema. Sin embargo, cualquier intento de descargar el módulo con `modprobe -r bluetooth` da como resultado el siguiente error:

   ```
   modprobe: FATAL: Module bluetooth is in use.
   ```

   ¿Cuál es la posible causa de este error?

Ejercicios Exploratorios

1. No es raro encontrar máquinas desactualizadas, obsoletas o mejor conocidas como legacy en entornos de producción, como cuando algunos equipos utilizan una conexión obsoleta para comunicarse con la computadora de control, por lo que es necesario prestar especial atención a algunas peculiaridades de estas máquinas más antiguas. Algunos servidores x86 con firmware BIOS más antiguo, por ejemplo, no se iniciarán si no se detecta un teclado. ¿Cómo se puede evitar este problema en particular?

2. Los sistemas operativos construidos alrededor del núcleo de Linux también están disponibles para una amplia variedad de arquitecturas de computadora que no sean x86, como en las computadoras de placa única basadas en la arquitectura ARM. Un usuario atento notará la ausencia del comando `lspci` en tales máquinas, como la Raspberry Pi. ¿Qué diferencia con las máquinas x86 justifica esa ausencia?

3. Muchos enrutadores de red tienen un puerto USB que permite las conexiones de un dispositivo externo, como un disco duro USB. Dado que la mayoría de estos utilizan un sistema operativo basado en Linux, ¿cómo se nombrará un disco duro USB externo en el directorio `/dev/`, suponiendo que no haya otro dispositivo de bloque convencional en el enrutador?

4. En 2018, se descubrió la vulnerabilidad de dispositivos conocida como *Meltdown*. Afecta a casi todos los procesadores de muchas arquitecturas. Las versiones recientes del núcleo de Linux pueden informar si el sistema actual es vulnerable. ¿Cómo se puede obtener esta información?

Resumen

Esta lección cubre los conceptos generales sobre cómo el núcleo de Linux maneja los recursos y dispositivos disponibles, principalmente en la arquitectura x86. La lección trata los siguientes temas:

- La configuración definida en las utilidades de configuración BIOS o UEFI puede afectar la forma en que el sistema operativo interactúa con los dispositivos.
- Cómo usar las herramientas proporcionadas por un sistema Linux estándar para obtener información sobre los dispositivos.
- Cómo identificar dispositivos de almacenamiento permanentes y extraíbles en el sistema de archivos.

Los comandos y procedimientos abordados fueron:

- Comandos para inspeccionar los dispositivos detectados: `lspci` y `lsusb`.
- Comandos para administrar módulos del núcleo del sistema operativo: `lsmod` y `modprobe`.
- Archivos especiales relacionados con los dispositivos, ya sean los archivos que se encuentran en el directorio `/dev/` o en los pseudo-sistemas de archivos en `/proc/` y `/sys/`.

Respuestas a los ejercicios guiados

1. Supongamos que un sistema operativo no puede iniciarse después de agregar un segundo disco SATA al sistema. Sabiendo que ninguno de los dispositivos está defectuoso, ¿cuál podría ser la posible causa de este error?

   El orden del dispositivo de arranque debe configurarse en la utilidad de configuración del BIOS; de lo contrario, es posible que el BIOS no pueda ejecutar el cargador de arranque.

2. Suponga que desea asegurarse de que la tarjeta de video externa conectada al bus PCI de su computadora de escritorio recién adquirida realmente sea la anunciada por el fabricante, pero al abrir el cajón de la computadora anularía la garantía. ¿Qué comando podría usarse para enumerar los detalles de la tarjeta de video, tal como fueron detectados por el sistema operativo?

   El comando `lspci` enumerará información detallada sobre todos los dispositivos actualmente conectados al bus PCI.

3. La siguiente línea es parte de la salida generada por el comando `lspci`:

   ```
   03:00.0 RAID bus controller: LSI Logic / Symbios Logic MegaRAID SAS 2208
   [Thunderbolt] (rev 05)
   ```

   ¿Qué comando debe ejecutar para identificar el módulo del núcleo de sistema operatico en uso para este dispositivo específico?

   El comando `lspci -s 03: 00.0 -v` o `lspci -s 03: 00.0 -k`

4. Un administrador del sistema quiere probar diferentes parámetros para el módulo del kernel `bluetooth` sin reiniciar el sistema. Sin embargo, cualquier intento de descargar el módulo con `modprobe -r bluetooth` da como resultado el siguiente error:

   ```
   modprobe: FATAL: Module bluetooth is in use.
   ```

   ¿Cuál es la posible causa de este error?

   El módulo `bluetooth` está siendo utilizado por un proceso en ejecución.

Respuestas a ejercicios exploratorios

1. No es raro encontrar máquinas desactualizadas, obsoletas o mejor conocidas como legacy en entornos de producción, como cuando algunos equipos utilizan una conexión obsoleta para comunicarse con la computadora de control, por lo que es necesario prestar especial atención a algunas peculiaridades de estas máquinas más antiguas. Algunos servidores x86 con firmware BIOS más antiguo, por ejemplo, no se iniciarán si no se detecta un teclado. ¿Cómo se puede evitar este problema en particular?

   La utilidad de configuración del BIOS tiene una opción para desactivar el bloqueo de la computadora cuando no se encuentra un teclado.

2. Los sistemas operativos construidos alrededor del núcleo de Linux también están disponibles para una amplia variedad de arquitecturas de computadora que no sean x86, como en las computadoras de placa única basadas en la arquitectura ARM. Un usuario atento notará la ausencia del comando `lspci` en tales máquinas, como la Raspberry Pi. ¿Qué diferencia con las máquinas x86 justifica esa ausencia?

   A diferencia de la mayoría de las máquinas x86, una computadora basada en ARM como la Raspberry Pi carece de un bus PCI, por lo que el comando `lspci` es inútil.

3. Muchos enrutadores de red tienen un puerto USB que permite las conexiones de un dispositivo externo, como un disco duro USB. Dado que la mayoría de estos utilizan un sistema operativo basado en Linux, ¿cómo se nombrará un disco duro USB externo en el directorio `/dev/`, suponiendo que no haya otro dispositivo de bloque convencional en el enrutador?

   Los kernels modernos de Linux identifican los discos duros USB como dispositivos SATA, por lo que el archivo correspondiente será `/dev/sda` ya que no existe ningún otro dispositivo de bloque convencional en el sistema.

4. En 2018, se descubrió la vulnerabilidad de dispositivos conocida como *Meltdown*. Afecta a casi todos los procesadores de muchas arquitecturas. Las versiones recientes del núcleo de Linux pueden informar si el sistema actual es vulnerable. ¿Cómo se puede obtener esta información?

   El archivo `/proc/cpuinfo` tiene una línea que muestra los errores conocidos de la CPU correspondiente, como `bugs: cpu_meltdown`.



101.2 Arranque del sistema

Referencia al objetivo del LPI

LPIC-1 v5, Exam 101, Objective 101.2

Importancia

3

Áreas de conocimiento clave

- Proporcionar comandos comunes al gestor de arranque y opciones al kernel en el momento del arranque.
- Demostrar conocimiento de la secuencia de arranque desde BIOS/UEFI hasta la finalización del arranque.
- Comprensión de SysVinit y systemd.
- Conocimiento de Upstart.
- Comprobar los eventos de arranque en los archivos de registro.

Lista parcial de archivos, términos y utilidades

- `dmesg`
- `journalctl`
- BIOS
- UEFI
- bootloader
- kernel
- `initramfs`

- `init`
- SysVinit
- systemd



101.2 Lección 1

Certificación:	LPIC-1
Versión:	5.0
Tema:	101 Arquitectura de Sistema
Objetivo:	101.2 Arranque de sistema
Lección:	1 de 1

Introducción

Para controlar la máquina, el componente principal del sistema operativo, el núcleo, debe cargarse mediante un programa llamado *bootloader*, que a su vez lo carga un firmware preinstalado como BIOS o UEFI. El gestor de arranque se puede personalizar para pasar parámetros al núcleo, como qué partición contiene el sistema de archivos raíz o en qué modo debe ejecutarse el sistema operativo. Una vez cargado, el núcleo continúa el proceso de arranque identificando y configurando el hardware. Por último, el núcleo del sistema operativo llama a la utilidad responsable de iniciar y administrar los servicios del sistema.

NOTE En algunas distribuciones de Linux, los comandos ejecutados en esta lección pueden requerir privilegios de root.

BIOS o UEFI

Los procedimientos ejecutados por las máquinas x86 para ejecutar el gestor de arranque son diferentes si usa BIOS o UEFI. El BIOS, abreviatura de *Basic Input/Output System*, es un programa almacenado en un chip de memoria no volátil conectado a la placa base, que se ejecuta cada vez que se enciende la computadora. Este tipo de programa se llama *firmware* y su ubicación de almacenamiento es independiente de los otros dispositivos de almacenamiento que pueda tener el sistema. El BIOS supone que los primeros 440 bytes en el primer dispositivo corresponden a almacenamiento, siguiendo el orden definido en la utilidad de configuración del BIOS, la primera etapa del cargador de arranque (también llamado *bootstrap*). Los primeros 512 bytes de un dispositivo de almacenamiento se denominan MBR (*Master Boot Record*), en dispositivos de almacenamiento que utilizan el esquema de partición estándar de DOS y, además de la primera etapa del gestor de arranque, contiene la tabla de particiones. Si el MBR no contiene los datos correctos, el sistema no podrá arrancar, a menos que se emplee un método alternativo.

En términos generales, los pasos previos a la operación para iniciar un sistema equipado con BIOS son:

1. El proceso POST (*power-on self-test*) se ejecuta para identificar fallas de dispositivos simples tan pronto como se enciende la máquina.
2. El BIOS activa los componentes básicos para cargar el sistema, como salida de video, teclado y medios de almacenamiento.
3. El BIOS carga la primera etapa del gestor de arranque desde el MBR (los primeros 440 bytes del primer dispositivo, como se define en la utilidad de configuración del BIOS).
4. La primera etapa del gestor de arranque llama a la segunda etapa del gestor de arranque, responsable de presentar las opciones de arranque y cargar el núcleo del sistema operativo.

El UEFI, abreviatura de *Unified Extensible Firmware Interface*, difiere del BIOS en algunos puntos claves. Como BIOS, el UEFI también es un firmware, pero puede identificar particiones y leer muchos sistemas de archivos que se encuentran en ellas. El UEFI no se basa en el MBR, teniendo en cuenta solo la configuración almacenada en su memoria no volátil (*NVRAM*) conectada a la placa base. Estas definiciones indican la ubicación de los programas compatibles con UEFI, llamados *EFI applications*, que se ejecutarán automáticamente o se llamarán desde un menú de arranque. Las aplicaciones EFI pueden ser gestores de arranque, selectores de sistema operativo, herramientas para el diagnóstico y reparación del sistema, etc. Deben estar en una partición de dispositivo de almacenamiento convencional y en un sistema de archivos compatible. Los sistemas de archivos compatibles estándar son FAT12, FAT16 y FAT32 para dispositivos de bloque e ISO-9660 para medios ópticos. Este enfoque permite la implementación de herramientas mucho más sofisticadas que las posibles con BIOS.

La partición que contiene las aplicaciones EFI se llama *EFI System Partition* o simplemente ESP. Esta partición no debe compartirse con otros sistemas de archivos del sistema, como el sistema de archivos raíz o los sistemas de archivos de datos del usuario. El directorio EFI en la partición ESP contiene las aplicaciones señaladas por las entradas guardadas en la NVRAM.

En términos generales, los pasos de arranque del sistema preoperativo en un sistema con UEFI son:

1. El proceso POST (*power-on self-test*) se ejecuta para identificar fallas de dispositivos simples tan pronto como se enciende la máquina.
2. El UEFI activa los componentes básicos para cargar el sistema, como salida de video, teclado y medios de almacenamiento.
3. El firmware de UEFI lee las definiciones almacenadas en NVRAM para ejecutar la aplicación EFI predefinida almacenada en el sistema de archivos de la partición ESP. Por lo general, la aplicación EFI predefinida es un gestor de arranque.
4. Si la aplicación EFI predefinida es un gestor de arranque, cargará el núcleo para iniciar el sistema operativo.

El estándar UEFI también admite una característica llamada *Secure Boot*, que solo permite la ejecución de aplicaciones EFI firmadas, es decir, aplicaciones EFI autorizadas por el fabricante del hardware. Esta característica aumenta la protección contra software malicioso, pero puede dificultar la instalación de sistemas operativos no cubiertos por la garantía del fabricante.

El Cargador de Arranque

El gestor de arranque más popular para Linux en la arquitectura x86 es GRUB (*Grand Unified Bootloader*). Tan pronto como lo llame el BIOS o el UEFI, GRUB muestra una lista de los sistemas operativos disponibles para arrancar. A veces, la lista no aparece automáticamente, pero se puede invocar presionando `Shift` mientras el BIOS está llamando a GRUB. En los sistemas UEFI, la tecla `Esc` debería usarse en su lugar.

Desde el menú GRUB es posible elegir cuál de los núcleos instalados debe cargarse y pasarle los parámetros. La mayoría de los parámetros del núcleo siguen el patrón `option=value`. Algunos de los parámetros del núcleo de Linux más útiles son:

`acpi`

Habilita/deshabilita el soporte ACPI. `acpi=off` deshabilitará la compatibilidad con ACPI.

`init`

Establece un iniciador de sistema alternativo. Por ejemplo, `init=/bin/bash` establecerá shell Bash

como iniciador. Esto significa que se iniciará una sesión de shell justo después del proceso de arranque del kernel.

`systemd.unit`

Establece a *systemd* para que se active. Por ejemplo, `systemd.unit=graphical.target`. Systemd también acepta los niveles de ejecución numéricos definidos para *SysV*. Para activar el nivel de ejecución 1, por ejemplo, solo es necesario incluir el número `1` o la letra `S` (abreviatura de "single") como parámetro del núcleo.

`mem`

Establece la cantidad de RAM disponible para el sistema. Este parámetro es útil cuando se utilizan máquinas virtuales, limitando la cantidad de RAM disponible para cada una de ellas. El uso de `mem=512M` limitará a 512 megabytes la cantidad de RAM disponible para un sistema virtual en particular.

`maxcpus`

Limita el número de procesadores (o núcleos de procesador) visibles para el sistema en máquinas simétricas multiprocesador. También es útil para máquinas virtuales. Un valor de `0` desactiva el soporte para máquinas multiprocesador y tiene el mismo efecto que el parámetro del núcleo `nosmp`. El parámetro `maxcpus=2` limitará el número de procesadores disponibles para el sistema operativo a dos.

`quiet`

Oculta la mayoría de los mensajes de arranque.

`vga`

Selecciona un modo de video. El parámetro `vga=ask` mostrará una lista de los modos disponibles para elegir.

`root`

Establece la partición raíz, distinta de la preconfigurada en el gestor de arranque. Por ejemplo, `root =/dev/sda3`.

`rootflags`

Opciones de montaje para el sistema de archivos raíz.

`ro`

Hace que el montaje inicial del sistema de archivos raíz sea de solo lectura.

`rw`

Permite escribir en el sistema de archivos raíz durante el montaje inicial.

Por lo general, no es necesario cambiar los parámetros del núcleo de Linux, pero puede ser útil para detectar y resolver problemas relacionados con el sistema operativo. Los parámetros del núcleo del sistema operativo deben agregarse al archivo `/etc/default/grub` en la línea `GRUB_CMDLINE_LINUX` para que sean persistentes durante los reinicios. Se debe generar un nuevo archivo de configuración para el gestor de arranque cada vez que `/etc/default/grub` cambie, lo cual se logra mediante el comando `grub-mkconfig -o /boot/grub/grub.cfg`. Una vez que el sistema operativo se está ejecutando, los parámetros del núcleo utilizados para cargar la sesión actual están disponibles en el archivo `/proc/cmdline`.

NOTE La configuración de GRUB se discutirá en próximas lecciones.

Inicialización del Sistema

Además del núcleo, el sistema operativo depende de otros componentes que proporcionan las características esperadas. Muchos de estos componentes se cargan durante el proceso de inicialización del sistema, que varía desde simples scripts de consola hasta programas de servicio más complejos. Las secuencias de comandos a menudo se utilizan para realizar tareas de corta duración que se ejecutarán y finalizarán durante el proceso de inicialización del sistema. Los servicios, también conocidos como demonios, pueden estar activos todo el tiempo, ya que pueden ser responsables de los aspectos intrínsecos del sistema operativo.

La diversidad de formas en que los scripts de inicio y los demonios con las características más diferentes se pueden integrar en una distribución de Linux es enorme, un hecho que históricamente obstaculizó el desarrollo de una solución única que cumpliera con las expectativas de los mantenedores y usuarios de todas las distribuciones de Linux. Sin embargo, cualquier herramienta que los encargados de la distribución hayan elegido para realizar esta función al menos podrá iniciar, detener y reiniciar los servicios del sistema. El sistema mismo suele realizar estas acciones después de una actualización de software, por ejemplo, pero el administrador del sistema casi siempre necesitará reiniciar manualmente el servicio después de realizar modificaciones en su archivo de configuración.

También es conveniente que un administrador del sistema pueda activar un conjunto particular de demonios, dependiendo de las circunstancias. Debería ser posible, por ejemplo, ejecutar solo un conjunto mínimo de servicios para realizar tareas de mantenimiento del sistema.

NOTE

Estrictamente hablando, el sistema operativo es solo el núcleo y sus componentes que controlan los dispositivos y gestionan todos los procesos. Sin embargo, es común usar el término "sistema operativo" de manera más flexible, para designar un grupo completo de programas distintos que componen el entorno de software donde el usuario puede realizar las tareas computacionales básicas.

La inicialización del sistema operativo comienza cuando el gestor de arranque carga el núcleo en la RAM. Luego, el núcleo se hará cargo de la CPU y comenzará a detectar y configurar los aspectos fundamentales del sistema operativo, como la configuración básica de los dispositivos y el direccionamiento de la memoria.

El núcleo del sistema operativo abrirá el *initramfs* (*initial RAM filesystem*). Initramfs es un archivo que contiene un sistema de archivos utilizado como un sistema de archivos raíz temporal durante el proceso de arranque. El objetivo principal de un archivo initramfs es proporcionar los módulos necesarios para que el núcleo pueda acceder al sistema de archivos raíz "real" del sistema operativo.

Tan pronto como el sistema de archivos raíz esté disponible, el núcleo montará todos los sistemas de archivos configurados en `/etc/fstab` y luego ejecutará el primer programa, una utilidad llamada `init`. El programa `init` es responsable de ejecutar todos los scripts de inicialización y demonios del sistema. Existen implementaciones distintas de tales iniciadores de sistemas aparte del init tradicional, como *systemd* y *Upstart*. Una vez que se carga el programa init, initramfs se elimina de la RAM.

SysV standard

Un administrador de servicios basado en el estándar SysVinit controla qué demonios y recursos estarán disponibles empleando el concepto de *runlevels*. Los niveles de ejecución están numerados del 0 al 6 y están diseñados por los encargados de la distribución para cumplir con propósitos específicos. Las únicas definiciones de nivel de ejecución compartidas entre todas las distribuciones son los niveles de ejecución 0, 1 y 6.

systemd

systemd es un administrador moderno de sistemas y servicios con una capa de compatibilidad para los comandos y niveles de ejecución de SysV. systemd tiene una estructura concurrente, emplea sockets y D-Bus para la activación del servicio, ejecución de demonios a demanda, monitoreo de procesos con *cgroups*, snapshot support, recuperación de sesión del sistema, control de punto de montaje y un control de servicio basado en la dependencia. En los últimos años, la mayoría de las principales distribuciones de Linux han adoptado gradualmente systemd como su administrador de sistema predeterminado.

Upstart

Al igual que systemd, Upstart es un sustituto de init. El objetivo de Upstart es acelerar el proceso de arranque paralelizando el proceso de carga de los servicios del sistema. Upstart fue utilizado por distribuciones basadas en Ubuntu en versiones anteriores, pero hoy dio paso a systemd.

Inspección de Inicialización

Pueden ocurrir errores durante el proceso de arranque, pero pueden no ser tan críticos para detener completamente el sistema operativo. No obstante, estos errores pueden comprometer el comportamiento esperado del sistema. Todos los errores dan como resultado mensajes que pueden usarse para futuras investigaciones, ya que contienen información valiosa sobre cuándo y cómo ocurrió el error. Incluso cuando no se generan mensajes de error, la información recopilada durante el proceso de arranque puede ser útil para fines de ajuste y configuración.

El espacio de memoria donde el kernel almacena sus mensajes, incluidos los mensajes de arranque, se llama *kernel ring buffer*. Los mensajes se guardan en el búfer del anillo del núcleo incluso cuando no se muestran durante el proceso de inicialización, como cuando se muestra una animación en su lugar. Sin embargo, el buffer del anillo del núcleo pierde todos los mensajes cuando el sistema está apagado o al ejecutar el comando `dmesg --clear`. Sin opciones, el comando `dmesg` muestra los mensajes actuales en el búfer del núcleo:

```
$ dmesg
[    5.262389] EXT4-fs (sda1): mounted filesystem with ordered data mode. Opts:
(null)
[    5.449712] ip_tables: (C) 2000-2006 Netfilter Core Team
[    5.460286] systemd[1]: systemd 237 running in system mode.
[    5.480138] systemd[1]: Detected architecture x86-64.
[    5.481767] systemd[1]: Set hostname to <torre>.
[    5.636607] systemd[1]: Reached target User and Group Name Lookups.
[    5.636866] systemd[1]: Created slice System Slice.
[    5.637000] systemd[1]: Listening on Journal Audit Socket.
[    5.637085] systemd[1]: Listening on Journal Socket.
[    5.637827] systemd[1]: Mounting POSIX Message Queue File System...
[    5.638639] systemd[1]: Started Read required files in advance.
[    5.641661] systemd[1]: Starting Load Kernel Modules...
[    5.661672] EXT4-fs (sda1): re-mounted. Opts: errors=remount-ro
[    5.694322] lp: driver loaded but no devices found
[    5.702609] ppdev: user-space parallel port driver
[    5.705384] parport_pc 00:02: reported by Plug and Play ACPI
[    5.705468] parport0: PC-style at 0x378 (0x778), irq 7, dma 3
[PCSPP,TRISTATE,COMPAT,EPP,ECP,DMA]
[    5.800146] lp0: using parport0 (interrupt-driven).
[    5.897421] systemd-journald[352]: Received request to flush runtime journal
from PID 1
```

La salida de `dmesg` puede tener cientos de líneas, por lo que la lista anterior contiene solo el extracto que muestra el núcleo que llama al administrador del servicio systemd. Los valores al comienzo de las líneas son la cantidad de segundos relativos al inicio de la carga del núcleo.

En sistemas basados en systemd, el comando `journalctl` mostrará los mensajes de inicialización con las opciones `-b`, `--boot`,`-k` o `--dmesg`. El comando `journalctl --list-boots` muestra una lista de números de arranque relativos al arranque actual, su hash de identificación y las marcas de tiempo del primer y último mensaje correspondientes:

```
$ journalctl --list-boots
 -4 9e5b3eb4952845208b841ad4dbefa1a6 Thu 2019-10-03 13:39:23 -03—Thu 2019-10-03
13:40:30 -03
 -3 9e3d79955535430aa43baa17758f40fa Thu 2019-10-03 13:41:15 -03—Thu 2019-10-03
14:56:19 -03
 -2 17672d8851694e6c9bb102df7355452c Thu 2019-10-03 14:56:57 -03—Thu 2019-10-03
19:27:16 -03
 -1 55c0d9439bfb4e85a20a62776d0dbb4d Thu 2019-10-03 19:27:53 -03—Fri 2019-10-04
00:28:47 -03
  0 08fbbebd9f964a74b8a02bb27b200622 Fri 2019-10-04 00:31:01 -03—Fri 2019-10-04
10:17:01 -03
```

Los registros de inicialización anteriores también se mantienen en sistemas basados en systemd, por lo que los mensajes de sesiones anteriores del sistema operativo aún se pueden inspeccionar. Si se proporcionan las opciones `-b 0` o `--boot=0`, se mostrarán los mensajes para el arranque actual. Las opciones `-b -1` o `--boot=-1` mostrarán mensajes de la inicialización anterior. Las opciones `-b -2` o `--boot=-2` mostrarán los mensajes de la inicialización antes de eso y así sucesivamente. El siguiente extracto muestra el llamado del sistema operativo al administrador del servicio systemd para el último proceso de inicialización:

```
$ journalctl -b 0
oct 04 00:31:01 ubuntu-host kernel: EXT4-fs (sda1): mounted filesystem with ordered
data mode. Opts: (null)
oct 04 00:31:01 ubuntu-host kernel: ip_tables: (C) 2000-2006 Netfilter Core Team
oct 04 00:31:01 ubuntu-host systemd[1]: systemd 237 running in system mode.
oct 04 00:31:01 ubuntu-host systemd[1]: Detected architecture x86-64.
oct 04 00:31:01 ubuntu-host systemd[1]: Set hostname to <torre>.
oct 04 00:31:01 ubuntu-host systemd[1]: Reached target User and Group Name Lookups.
oct 04 00:31:01 ubuntu-host systemd[1]: Created slice System Slice.
oct 04 00:31:01 ubuntu-host systemd[1]: Listening on Journal Audit Socket.
oct 04 00:31:01 ubuntu-host systemd[1]: Listening on Journal Socket.
oct 04 00:31:01 ubuntu-host systemd[1]: Mounting POSIX Message Queue File System...
oct 04 00:31:01 ubuntu-host systemd[1]: Started Read required files in advance.
oct 04 00:31:01 ubuntu-host systemd[1]: Starting Load Kernel Modules...
oct 04 00:31:01 ubuntu-host kernel: EXT4-fs (sda1): re-mounted. Opts:
commit=300,barrier=0,errors=remount-ro
oct 04 00:31:01 ubuntu-host kernel: lp: driver loaded but no devices found
oct 04 00:31:01 ubuntu-host kernel: ppdev: user-space parallel port driver
oct 04 00:31:01 ubuntu-host kernel: parport_pc 00:02: reported by Plug and Play
ACPI
oct 04 00:31:01 ubuntu-host kernel: parport0: PC-style at 0x378 (0x778), irq 7, dma
3 [PCSPP,TRISTATE,COMPAT,EPP,ECP,DMA]
oct 04 00:31:01 ubuntu-host kernel: lp0: using parport0 (interrupt-driven).
oct 04 00:31:01 ubuntu-host systemd-journald[352]: Journal started
oct 04 00:31:01 ubuntu-host systemd-journald[352]: Runtime journal
(/run/log/journal/abb765408f3741ae9519ab3b96063a15) is 4.9M, max 39.4M, 34.5M free.
oct 04 00:31:01 ubuntu-host systemd-modules-load[335]: Inserted module 'lp'
oct 04 00:31:01 ubuntu-host systemd-modules-load[335]: Inserted module 'ppdev'
oct 04 00:31:01 ubuntu-host systemd-modules-load[335]: Inserted module 'parport_pc'
oct 04 00:31:01 ubuntu-host systemd[1]: Starting Flush Journal to Persistent
Storage...
```

La inicialización y otros mensajes emitidos por el sistema operativo se almacenan en archivos dentro del directorio `/var/log/`. Si ocurre un error crítico y el sistema operativo no puede continuar el proceso de inicialización después de cargar el kernel y el initramfs, se podría usar un medio de arranque alternativo para iniciar el sistema y acceder al sistema de archivos correspondiente. Luego, los archivos almacendos en `/var/log/` pueden buscarse por posibles razones que causan la interrupción del proceso de arranque. Las opciones `-D` o `--directory` del comando `journalctl` se pueden usar para leer mensajes de registro en directorios que no sean `/var/log/journal/`, que es la ubicación predeterminada para los mensajes de registro de systemd. Como los mensajes de registro de systemd se almacenan en texto sin formato, se requiere el comando `journalctl` para leerlos.

Ejercicios Guiados

1. En una máquina equipada con un firmware de BIOS, ¿dónde se encuentra el binario de arranque?

2. El firmware UEFI admite funciones ampliadas proporcionadas por programas externos, llamadas aplicaciones EFI. Estas aplicaciones, sin embargo, tienen su propia ubicación especial. ¿En qué parte del sistema se ubicarían las aplicaciones EFI?

3. Los cargadores de arranque permiten pasar parámetros personalizados del kernel antes de cargarlo. Suponga que el sistema no puede iniciarse debido a una ubicación errónea del sistema de archivos raíz. ¿Cómo se le daría al kernel el sistema de archivos raíz correcto, ubicado en `/dev/sda3`?

4. El proceso de arranque de una máquina Linux termina con el siguiente mensaje:

```
ALERT! /dev/sda3 does not exist. Dropping to a shell!
```

   ¿Cuál es la causa probable de este problema?

Ejercicios Exploratorios

1. El gestor de arranque presentará una lista de sistemas operativos para elegir cuando se instala más de un sistema operativo en la máquina. Sin embargo, un sistema operativo recién instalado puede sobrescribir el MBR del disco duro, borrando la primera etapa del gestor de arranque y haciendo que el otro sistema operativo sea inaccesible. ¿Por qué esto no sucedería en una máquina equipada con un firmware UEFI?

2. ¿Cuál es una consecuencia común de instalar un kernel personalizado sin proporcionar una imagen initramfs adecuada?

3. El registro de inicialización tiene cientos de líneas, por lo que la salida del comando `dmesg` a menudo se canaliza a un método de lectura más ágil, como el comando `less`. ¿Qué opción `dmesg` paginará automáticamente su salida, eliminando la necesidad de usar un comando de paginación explícitamente?

4. Un disco duro que contiene todo el sistema de archivos de una máquina desconectado se removió y se conectó a una máquina en funcionamiento como unidad secundaria. Suponiendo que su punto de montaje es `/mnt/hd`, ¿cómo se usaría `journalctl` para inspeccionar el contenido de los archivos de logs ubicados en `/mnt/hd/var/log/journal/`?

Resumen

Esta lección cubre la secuencia de arranque en un sistema Linux estándar. El conocimiento adecuado de cómo funciona el proceso de arranque de un sistema Linux ayuda a evitar errores que pueden hacer que el sistema sea inaccesible. La lección trata los siguientes temas:

- ¿Cómo difieren los métodos de arranque BIOS y UEFI?
- Etapas típicas de inicialización del sistema.
- Mensajes de recuperación de arranque.

Los comandos y procedimientos abordados fueron:

- Parámetros comunes del núcleo del sistema operativo.
- Comandos para leer mensajes de arranque: `dmesg` y `journalctl`.

Respuestas a los ejercicios guiados

1. En una máquina equipada con un firmware de BIOS, ¿dónde se encuentra el binario de arranque?

   En el MBR del primer dispositivo de almacenamiento, como se define en la utilidad de configuración del BIOS.

2. El firmware UEFI admite funciones ampliadas proporcionadas por programas externos, llamadas aplicaciones EFI. Estas aplicaciones, sin embargo, tienen su propia ubicación especial. ¿En qué parte del sistema se ubicarían las aplicaciones EFI?

   Las aplicaciones EFI se almacenan en la partición del sistema EFI (ESP), ubicada en cualquier bloque de almacenamiento disponible con un sistema de archivos compatible (generalmente un sistema de archivos FAT32).

3. Los cargadores de arranque permiten pasar parámetros personalizados del kernel antes de cargarlo. Suponga que el sistema no puede iniciarse debido a una ubicación errónea del sistema de archivos raíz. ¿Cómo se le daría al kernel el sistema de archivos raíz correcto, ubicado en `/dev/sda3`?

   Se debe usar el parámetro `root`, como `root=/dev/sda3`.

4. El proceso de arranque de una máquina Linux termina con el siguiente mensaje:

   ```
   ALERT! /dev/sda3 does not exist. Dropping to a shell!
   ```

   ¿Cuál es la causa probable de este problema?

   El núcleo no pudo encontrar el dispositivo `/dev/sda3`, determinado como el sistema de archivos raíz.

Respuestas a ejercicios exploratorios

1. El gestor de arranque presentará una lista de sistemas operativos para elegir cuando se instala más de un sistema operativo en la máquina. Sin embargo, un sistema operativo recién instalado puede sobrescribir el MBR del disco duro, borrando la primera etapa del gestor de arranque y haciendo que el otro sistema operativo sea inaccesible. ¿Por qué esto no sucedería en una máquina equipada con un firmware UEFI?

   Las máquinas UEFI no usan el MBR del disco duro para almacenar la primera etapa del gestor de arranque.

2. ¿Cuál es una consecuencia común de instalar un kernel personalizado sin proporcionar una imagen initramfs adecuada?

   El sistema de archivos raíz puede ser inaccesible si fue compilado como un módulo de kernel externo.

3. El registro de inicialización tiene cientos de líneas, por lo que la salida del comando `dmesg` a menudo se canaliza a un método de lectura más ágil, como el comando `less`. ¿Qué opción `dmesg` paginará automáticamente su salida, eliminando la necesidad de usar un comando de paginación explícitamente?

   Los comandos `dmesg -H` o `dmesg --human` habilitarán el paginador por defecto.

4. Un disco duro que contiene todo el sistema de archivos de una máquina desconectado se removió y se conectó a una máquina en funcionamiento como unidad secundaria. Suponiendo que su punto de montaje es `/mnt/hd`, ¿cómo se usaría `journalctl` para inspeccionar el contenido de los archivos de logs ubicados en `/mnt/hd/var/log/journal/`?

   Con los comandos `journalctl -D /mnt/hd/var/log/journal` o `journalctl --directory=/mnt/hd/var/log/journal`



101.3 Cambiar los niveles de ejecución / objetivos de arranque y apagar o reiniciar el sistema

Referencia al objetivo del LPI

LPIC-1 v5, Exam 101, Objective 101.3

Importancia

3

Áreas de conocimiento clave

- Establecer el nivel de ejecución o el objetivo de arranque predeterminado.
- * Cambiar entre niveles de ejecución / objetivos de arranque, incluido el modo monousuario. Apagar y reiniciar desde la línea de comandos.
- Alertar a los usuarios antes de cambiar de nivel de ejecución/objetivo de arranque u otros eventos importantes del sistema.
- Terminar procesos de forma adecuada.
- Conocimiento de acpid.

Lista parcial de archivos, términos y utilidades

- `/etc/inittab`
- `shutdown`
- `init`
- `/etc/init.d/`
- `telinit`
- `systemd`

- `systemctl`
- `/etc/systemd/`
- `/usr/lib/systemd/`
- `wall`



101.3 Lección 1

Certificación:	LPIC-1
Versión:	5.0
Tema:	101 Arquitectura de Sistema
Objetivo:	101.3 Cambiar los niveles de ejecución / objetivos de arranque y apagar o reiniciar el sistema
Lección:	1 de 1

Introducción

Una característica común entre los sistemas operativos que siguen los principios de diseño de Unix es el empleo de procesos separados para controlar distintas funciones del sistema. Estos procesos, llamados *daemons* (o, más generalmente, *services*), también son responsables de las características extendidas subyacentes al sistema operativo, como los servicios de aplicaciones de red (servidor HTTP, intercambio de archivos, correo electrónico, etc.), bases de datos, configuración bajo demanda, etc. Aunque Linux utiliza un núcleo monolítico, muchos aspectos de bajo nivel del sistema operativo se ven afectados por demonios, como el equilibrio de carga y la configuración del firewall.

Los demonios (*daemons*) que deberían estar activos dependen del propósito del sistema. El conjunto de demonios activos también debe poder modificarse en tiempo de ejecución, de modo que los servicios se puedan iniciar y detener sin tener que reiniciar todo el sistema. Para abordar este problema, cada distribución principal de Linux ofrece alguna forma o utilidad de administración de servicios para controlar el sistema.

Los servicios pueden ser controlados por scripts de shell o por un programa y sus archivos de configuración compatibles. El primer método lo implementa el estándar *SysVinit*, también conocido

como *System V* o simplemente *SysV*. El segundo método es implementado por *systemd* y *Upstart*. Históricamente, los administradores de servicios basados en SysV fueron los más utilizados por las distribuciones de Linux. Hoy en día, los administradores de servicios basados en sistemas se encuentran con mayor frecuencia en la mayoría de las distribuciones de Linux. El administrador de servicios es el primer programa lanzado por el núcleo durante el proceso de arranque, por lo que su PID (número de identificación del proceso) siempre es `1`.

Estándar SysVinit

Un administrador de servicios basado en el estándar SysVinit proporcionará conjuntos predefinidos de estados del sistema, llamados *runlevels*, y sus correspondientes archivos de script de servicio para ser ejecutados. Los niveles de ejecución están numerados de `0` a `6`, y generalmente se asignan a los siguientes propósitos:

Runlevel 0

Apagado del sistema.

Runlevel 1, s o usuario único

Modo de usuario único, sin red y otras capacidades no esenciales (modo de mantenimiento).

Runlevel 2, 3 o 4

Modo multiusuario. Los usuarios pueden iniciar sesión por consola o red. Los niveles de ejecución 2 y 4 no se usan con frecuencia.

Runlevel 5

Modo multiusuario. Es equivalente a 3, más el inicio de sesión en modo gráfico.

Runlevel 6

Reinicio del sistema.

El programa responsable de administrar los niveles de ejecución y los demonios/recursos asociados es `/sbin/init`. Durante la inicialización del sistema, el programa `init` identifica el nivel de ejecución solicitado, definido por un parámetro del núcleo del sistema operativo o en el archivo `/etc/inittab`, y carga los scripts asociados que se enumeran allí para el nivel de ejecución dado. Cada nivel de ejecución puede tener muchos archivos de servicio asociados, generalmente scripts en el directorio `/etc/init.d/`. Como no todos los niveles de ejecución son equivalentes a través de diferentes distribuciones de Linux, también se puede encontrar una breve descripción del propósito del nivel de ejecución en las distribuciones basadas en SysV.

La sintaxis del archivo `/etc/inittab` usa este formato:

```
id:runlevels:action:process
```

El `id` es un nombre genérico de hasta cuatro caracteres de longitud utilizado para identificar la entrada. La entrada `runlevels` es una lista de números de niveles para los que se debe ejecutar una acción específica. El término `action` define cómo `init` ejecutará el proceso indicado por el término `process`. Las acciones disponibles son:

`boot`
: El proceso se ejecutará durante la inicialización del sistema. El campo `runlevels` se ignora.

`bootwait`
: El proceso se ejecutará durante la inicialización del sistema e `init` esperará hasta que termine para continuar. El campo `runlevels` se ignora.

`sysinit`
: El proceso se ejecutará después de la inicialización del sistema, independientemente del nivel de ejecución. El campo `runlevels` se ignora.

`wait`
: El proceso se ejecutará para los niveles de ejecución dados e `init` esperará hasta que termine para continuar.

`respawn`
: El proceso se reiniciará si finaliza.

`ctrlaltdel`
: El proceso se ejecutará cuando el proceso init reciba la señal `SIGINT`, que se activará cuando se presione la secuencia de teclas Ctrl + Alt + Supr.

El nivel de ejecución predeterminado, el que se elegirá si no se proporciona otro como parámetro del núcleo, también se define en `/etc/inittab`, en la entrada `id:x:initdefault`. La `x` es el número del nivel de ejecución predeterminado. Este número nunca debe ser `0` o `6`, ya que provocaría que el sistema se apague o reinicie tan pronto como finalice el proceso de arranque. A continuación se muestra un archivo típico `/etc/inittab`:

```
# Nivel de ejecución predeterminado
id:3:initdefault:

# Script de configuración ejecutado durante el arranque
si::sysinit:/etc/init.d/rcS

# Acción tomada en el nivel de ejecución S (usuario único)
~:S:wait:/sbin/sulogin

# Configuración para cada nivel de ejecución
l0:0:wait:/etc/init.d/rc 0
l1:1:wait:/etc/init.d/rc 1
l2:2:wait:/etc/init.d/rc 2
l3:3:wait:/etc/init.d/rc 3
l4:4:wait:/etc/init.d/rc 4
l5:5:wait:/etc/init.d/rc 5
l6:6:wait:/etc/init.d/rc 6

# Acción tomada sobre teclado ctrl+alt+del
ca::ctrlaltdel:/sbin/shutdown -r now

# Habilitar consolas para los niveles de ejecución 2 y 3
1:23:respawn:/sbin/getty tty1 VC linux
2:23:respawn:/sbin/getty tty2 VC linux
3:23:respawn:/sbin/getty tty3 VC linux
4:23:respawn:/sbin/getty tty4 VC linux

# Para el nivel de ejecución 3, también habilite serial
# terminales ttyS0 y ttyS1 (módem) consolas
S0:3:respawn:/sbin/getty -L 9600 ttyS0 vt320
S1:3:respawn:/sbin/mgetty -x0 -D ttyS1
```

El comando `telinit q` debe ejecutarse cada vez que se modifica el archivo `/etc/inittab`. El argumento `q` (o `Q`) le dice a init que vuelva a cargar su configuración. Este paso es importante para evitar que el sistema se detenga debido a una configuración incorrecta en `/etc/inittab`.

Los scripts utilizados por `init` para configurar cada nivel de ejecución se almacenan en el directorio `/etc/init.d/`. Cada nivel de ejecución tiene un directorio asociado en `/etc/`, llamado `/etc/rc0.d/`, `/etc/rc1.d/`, `/etc/rc2.d/`, etc., con los scripts eso debería ejecutarse cuando comience el nivel de ejecución correspondiente. Como el mismo script puede ser usado por diferentes niveles de ejecución, los archivos en esos directorios son solo enlaces simbólicos a los scripts reales en `/etc/init.d/`. Además, la primera letra del nombre de archivo del enlace en el directorio del nivel de ejecución indica si el servicio debe iniciarse o terminarse para el nivel de ejecución correspondiente.

El nombre de archivo de un enlace que comienza con la letra `K` determina que el servicio se eliminará al ingresar el nivel de ejecución (kill). Comenzando con la letra `S`, el servicio se iniciará al ingresar el nivel de ejecución (inicio). El directorio `/etc/rc1.d/`, por ejemplo, tendrá muchos enlaces a los scripts de red que comienzan con la letra `K`, considerando que el nivel de ejecución 1 corresponde a un solo usuario, sin conectividad de red.

El comando `runlevel` muestra el nivel de ejecución actual para el sistema. El comando `runlevel` muestra dos valores, el primero es el nivel de ejecución anterior y el segundo es el nivel de ejecución actual:

```
$ runlevel
N 3
```

La letra `N` en la salida muestra que el nivel de ejecución no ha cambiado desde el último arranque. En el ejemplo, el `runlevel 3` es el nivel de ejecución actual del sistema.

El mismo programa `init` puede usarse para alternar entre niveles de ejecución en un sistema en ejecución, sin la necesidad de reiniciar. El comando `telinit` también se puede usar para alternar entre los niveles de ejecución. Por ejemplo, los comandos `telinit 1`,`telinit s` o `telinit S` cambiarán el sistema al nivel de ejecución 1.

systemd

Actualmente, systemd es el conjunto de herramientas más utilizado para administrar los recursos y servicios del sistema, que systemd denomina unidades (*units*). Una unidad consta de nombre, tipo y un archivo de configuración correspondiente. Por ejemplo, la unidad para un proceso de servidor *httpd* (como el servidor web Apache) será `httpd.service` en distribuciones basadas en Red Hat, su archivo de configuración también se llamará `httpd.service` (en distribuciones basadas en Debian esta unidad es llamada `apache2.service`).

Hay siete tipos distintos de unidades systemd:

`service`
: El tipo de unidad más común, para recursos activos del sistema que se pueden iniciar, interrumpir y recargar.

`socket`
: El tipo de unidad de socket puede ser un socket de sistema de archivos o un socket de red. Todas las unidades de socket tienen una unidad de servicio correspondiente, cargada cuando el socket recibe una solicitud.

`device`
: Una unidad de dispositivo está asociada con un dispositivo de hardware identificado por el núcleo. Un dispositivo solo se tomará como una unidad systemd si existe una regla udev para este propósito. Se puede usar una unidad de dispositivo para resolver dependencias de configuración cuando se detecta cierto hardware, dado que las propiedades de la regla udev se pueden usar como parámetros para la unidad de dispositivo.

`mount`
: Una unidad de montaje es una definición de punto de montaje en el sistema de archivos, similar a una entrada en `/etc/fstab`.

`automount`
: Una unidad de montaje automático también es una definición de punto de montaje en el sistema de archivos, pero se monta automáticamente. Cada unidad de montaje automático tiene una unidad de montaje correspondiente, que se inicia cuando se accede al punto de montaje automático.

`target`
: Una unidad target es una agrupación de otras unidades, administradas como una sola unidad.

`snapshot`
: Una unidad snapshot es un estado guardado del administrador del sistema (no disponible en todas las distribuciones de Linux).

El comando principal para controlar las unidades systemd es `systemctl`. El comando `systemctl` se usa para ejecutar todas las tareas relacionadas con la activación, desactivación, ejecución, interrupción, monitoreo de la unidad, etc. Para una unidad ficticia llamada `unit.service`, por ejemplo, las acciones más comunes de `systemctl` serán:

`systemctl start unit.service`
: Inicia una unidad (`unit`).

`systemctl stop unit.service`
: Detiene una unidad (`unit`).

`systemctl restart unit.service`
: Reinicia una unidad (`unit`).

`systemctl status unit.service`
Muestra el estado de la unidad (`unit`), incluyendo si está en ejecución o no.

`systemctl is-active unit.service`
Muestra *active* Si la unidad (`unit`) está en ejecución o inactiva.

`systemctl enable unit.service`
La unidad (`unit`) se cargará durante la inicialización del sistema.

`systemctl disable unit.service`
La unidad (`unit`) no se cargará durante la inicialización del sistema.

`systemctl is-enabled unit.service`
Verifica si la unidad (`unit`) comienza con el sistema. La respuesta se almacena en la variable `$?`. El valor `0` indica que `unit` comienza con el sistema y el valor `1` indica que `unit` no comienza con el sistema.

NOTE

Las instalaciones más recientes de systemd en realidad enumerarán la configuración de una unidad para el tiempo de arranque. Por ejemplo:

```
$ systemctl is-enabled apparmor.service
enabled
```

Si no existen otras unidades con el mismo nombre en el sistema, el sufijo después del punto se puede quitar. Si, por ejemplo, solo hay una unidad `httpd` de tipo `service`, entonces solo `httpd` es suficiente como parámetro de unidad para `systemctl`.

El comando `systemctl` también puede controlar *system targets*. La unidad `multi-user.target`, por ejemplo, combina todas las unidades requeridas por el entorno del sistema multiusuario. Es similar al nivel de ejecución número 3 en un sistema que utiliza SysV.

El comando `systemctl isolate` alterna entre diferentes objetivos. Por lo tanto, para alternar manualmente al objetivo `multiusuario`:

```
# systemctl isolate multi-user.target
```

Hay objetivos correspondientes a los niveles de ejecución de SysV, comenzando con `runlevel0.target` hasta `runlevel6.target`. Sin embargo, systemd no utiliza el archivo

`/etc/inittab`. Para cambiar el objetivo predeterminado del sistema, la opción `systemd.unit` se puede agregar a la lista de parámetros del núcleo del sistema operativo. Por ejemplo, para usar `multi-user.target` como objetivo estándar, el parámetro del núcleo del sistema operativo debe ser `systemd.unit=multi-user.target`. Todos los parámetros del kernel pueden hacerse persistentes cambiando la configuración del gestor de arranque.

Otra forma de cambiar el objetivo predeterminado es modificar el enlace simbólico `/etc/systemd/system/default.target` para que apunte al objetivo deseado. La redefinición del enlace se puede hacer con el comando `systemctl` por sí mismo:

```
# systemctl set-default multi-user.target
```

Del mismo modo, puede determinar cuál es el objetivo de arranque predeterminado de su sistema con el siguiente comando:

```
$ systemctl get-default
graphical.target
```

Similar a los sistemas que adoptan SysV, el objetivo predeterminado nunca debe apuntar a `shutdown.target`, ya que corresponde al nivel de ejecución 0 (apagado).

Los archivos de configuración asociados con cada unidad se pueden encontrar en el directorio `/lib/systemd/system/`. El comando `systemctl list-unit-files` enumera todas las unidades disponibles y muestra si están habilitadas para iniciarse cuando se inicia el sistema. La opción `--type` seleccionará solo las unidades para un tipo dado, como en `systemctl list-unit-files --type=service` y `systemctl list-unit-files --type=target`.

Las unidades activas o unidades que han estado activas durante la sesión actual del sistema se pueden enumerar con el comando `systemctl list-units`. Al igual que la opción `list-unit-files`, el comando `systemctl list-units --type=service` seleccionará solo unidades de tipo `service` y el comando `systemctl list-units --type=target` seleccionará solo unidades de tipo `target`.

systemd también es responsable de desencadenar y responder a eventos relacionados con la energía del sistema. El comando `systemctl suspend` pondrá el sistema en modo de bajo consumo, manteniendo los datos actuales en la memoria. El comando `systemctl hibernate` copiará todos los datos de la memoria al disco, por lo que el estado actual del sistema se puede recuperar después de apagarlo. Las acciones asociadas con tales eventos se definen en el archivo `/etc/systemd/logind.conf` o en archivos separados dentro del directorio `/etc/systemd/logind.conf.d/`. Sin embargo, esta función systemd solo se puede usar cuando no hay otro administrador de energía ejecutándose en el sistema, como el demonio `acpid`. El demonio `acpid`

es el principal administrador de energía para Linux y permite ajustes más precisos a las acciones posteriores a eventos relacionados con la energía, como cerrar la tapa del portátil, batería baja o niveles de carga de la batería.

Upstart

Los scripts de inicialización utilizados por Upstart se encuentran en el directorio `/etc/init/`. Los servicios del sistema se pueden enumerar con el comando `initctl list`, que también muestra el estado actual de los servicios y, si está disponible, su número PID.

```
# initctl list
avahi-cups-reload stop/waiting
avahi-daemon start/running, process 1123
mountall-net stop/waiting
mountnfs-bootclean.sh start/running
nmbd start/running, process 3085
passwd stop/waiting
rc stop/waiting
rsyslog start/running, process 1095
tty4 start/running, process 1761
udev start/running, process 1073
upstart-udev-bridge start/running, process 1066
console-setup stop/waiting
irqbalance start/running, process 1842
plymouth-log stop/waiting
smbd start/running, process 1457
tty5 start/running, process 1764
failsafe stop/waiting
```

Cada acción de Upstart tiene su propio comando independiente. Por ejemplo, el comando `start` puede usarse para iniciar la sexta terminal virtual:

```
# start tty6
```

El estado actual de un recurso se puede verificar con el comando `status`:

```
# status tty6
tty6 start/running, process 3282
```

Y la interrupción de un servicio se realiza con el comando `stop`:

```
# stop tty6
```

Upstart no usa el archivo `/etc/inittab` para definir los niveles de ejecución, pero los comandos heredados `runlevel` y `telinit` todavía pueden usarse para verificar y alternar entre los niveles de ejecución.

> **NOTE** Upstart fue desarrollado para la distribución Ubuntu Linux para ayudar a facilitar el inicio paralelo de los procesos. Ubuntu ha dejado de usar Upstart desde 2015 cuando cambió de Upstart a systemd.

Apagado y Reinicio

Un comando muy tradicional utilizado para apagar o reiniciar el sistema se llama `shutdown`. El comando `shutdown` agrega funciones adicionales al proceso de apagado: notifica automáticamente a todos los usuarios conectados con un mensaje de advertencia en sus sesiones de shell y se evitan nuevos inicios de sesión. El comando `shutdown` actúa como intermediario para los procedimientos SysV o systemd, es decir, ejecuta la acción solicitada llamando a la acción correspondiente en el administrador de servicios adoptado por el sistema.

Después de ejecutar `shutdown`, todos los procesos reciben la señal `SIGTERM`, seguida de la señal `SIGKILL`, luego el sistema se apaga o cambia su nivel de ejecución. Por defecto, cuando no se utilizan las opciones `-h` o `-r`, el sistema alterna al nivel de ejecución 1, es decir, el modo de usuario único. Para cambiar las opciones predeterminadas para `shutdown`, el comando debe ejecutarse con la siguiente sintaxis:

```
$ shutdown [option] time [message]
```

Solo se requiere el parámetro `time`. El parámetro `time` define cuándo se ejecutará la acción solicitada, aceptando los siguientes formatos:

`hh:mm`
: Este formato especifica el tiempo de ejecución como hora y minutos.

`+m`
: Este formato especifica cuántos minutos esperar antes de la ejecución.

`now` o `+0`
: Este formato determina la ejecución inmediata.

El parámetro `message` es el texto de advertencia enviado a todas las sesiones de terminal de los usuarios conectados.

La implementación de SysV permite limitar a los usuarios que podrán reiniciar la máquina presionando Ctrl + Alt + Supr. Esto es posible colocando la opción `-a` para el comando `shutdown` presente en la línea con respecto a `ctrlaltdel` en el archivo `/etc/inittab`. Al hacer esto, solo los usuarios cuyos nombres de usuario estén en el archivo `/etc/shutdown.allow` podrán reiniciar el sistema con la combinación de teclas Ctrl + Alt + Supr.

El comando `systemctl` también se puede usar para apagar o reiniciar la máquina en sistemas que emplean systemd. Para reiniciar el sistema, se debe usar el comando `systemctl reboot`. Para apagar el sistema, se debe usar el comando `systemctl poweroff`. Ambos comandos requieren privilegios de root para ejecutarse, ya que los usuarios comunes no pueden realizar dichos procedimientos.

NOTE

Algunas distribuciones de Linux vincularán `poweroff` y `reboot` a `systemctl` como comandos individuales. Por ejemplo:

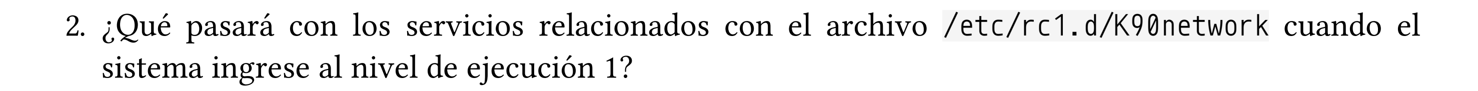
```
$ sudo which poweroff
/usr/sbin/poweroff
$ sudo ls -l /usr/sbin/poweroff
lrwxrwxrwx 1 root root 14 Aug 20 07:50 /usr/sbin/poweroff ->
/bin/systemctl
```

No todas las actividades de mantenimiento requieren que el sistema se apague o reinicie. Sin embargo, cuando es necesario cambiar el estado del sistema al modo de usuario único, es importante advertir a los usuarios que han iniciado sesión para que no se vean perjudicados por la finalización brusca de sus actividades.

De manera similar a lo que hace el comando `shutdown` cuando se apaga o reinicia el sistema, el comando `wall` puede enviar un mensaje a las sesiones de terminal de todos los usuarios conectados. Para hacerlo, el administrador del sistema solo necesita proporcionar un archivo o escribir directamente el mensaje como parámetro para ordenar `wall`.

Ejercicios Guiados

1. ¿Cómo podría usarse el comando `telinit` para reiniciar el sistema?



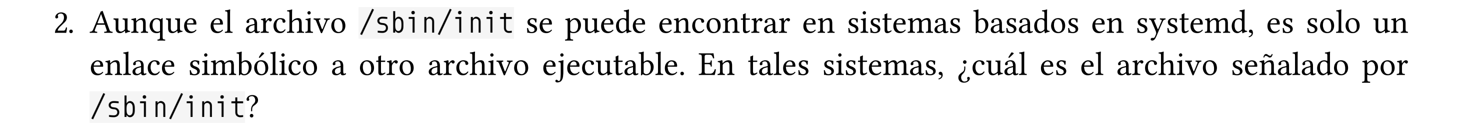
2. ¿Qué pasará con los servicios relacionados con el archivo `/etc/rc1.d/K90network` cuando el sistema ingrese al nivel de ejecución 1?

3. Usando el comando `systemctl`, ¿cómo podría un usuario verificar si la unidad `sshd.service` está funcionando?

4. En un sistema basado en systemd, ¿qué comando debe ejecutarse para permitir la activación de la unidad `sshd.service` durante la inicialización del sistema?

Ejercicios Exploratorios

1. En un sistema basado en SysV, suponga que el nivel de ejecución predeterminado definido en `/etc/inittab` es 3, pero el sistema siempre comienza en el nivel de ejecución 1. ¿Cuál es la causa probable de eso?



2. Aunque el archivo `/sbin/init` se puede encontrar en sistemas basados en systemd, es solo un enlace simbólico a otro archivo ejecutable. En tales sistemas, ¿cuál es el archivo señalado por `/sbin/init`?

3. ¿Cómo se puede verificar el objetivo predeterminado del sistema (system_target) en un sistema basado en systemd?

4. ¿Cómo se puede cancelar un reinicio del sistema programado con el comando `shutdown`?

Resumen

Esta lección cubre las principales utilidades utilizadas como administradores de servicios por las distribuciones de Linux. Las utilidades SysVinit, systemd y Upstart tienen su propio enfoque para controlar los servicios del sistema y los estados del sistema. La lección trata los siguientes temas:

- ¿Qué son los servicios del sistema y su papel en el sistema operativo?
- Conceptos y uso básico de los comandos SysVinit, systemd y Upstart.
- ¿Cómo iniciar, detener y reiniciar correctamente los servicios del sistema y el sistema mismo?

Los comandos y procedimientos abordados fueron:

- Comandos y archivos relacionados con SysVinit, como `init`, `/etc/inittab` y `telinit`.
- El comando principal de systemd: `systemctl`.
- Comandos de inicio: `initctl`, `status`, `start`, `stop`.
- Comandos tradicionales de administración de energía, como `shutdown` y `wall`.

Respuestas a los ejercicios guiados

1. ¿Cómo podría usarse el comando `telinit` para reiniciar el sistema?

   El comando `telinit 6` se alternará con el nivel de ejecución 6, es decir, reiniciar el sistema.

2. ¿Qué pasará con los servicios relacionados con el archivo `/etc/rc1.d/K90network` cuando el sistema ingrese al nivel de ejecución 1?

   Debido a la letra `K` al comienzo del nombre del archivo, los servicios relacionados se detendrán.

3. Usando el comando `systemctl`, ¿cómo podría un usuario verificar si la unidad `sshd.service` está funcionando?

   Con el comando `systemctl status sshd.service` o `systemctl is-active sshd.service`.

4. En un sistema basado en systemd, ¿qué comando debe ejecutarse para permitir la activación de la unidad `sshd.service` durante la inicialización del sistema?

   Comando `systemctl enable sshd.service`, ejecutado por root.

Respuestas a ejercicios exploratorios

1. En un sistema basado en SysV, suponga que el nivel de ejecución predeterminado definido en `/etc/inittab` es 3, pero el sistema siempre comienza en el nivel de ejecución 1. ¿Cuál es la causa probable de eso?

   Los parámetros `1` o `S` pueden estar presentes en la lista de parámetros del núcleo del sistema operativo.

2. Aunque el archivo `/sbin/init` se puede encontrar en sistemas basados en systemd, es solo un enlace simbólico a otro archivo ejecutable. En tales sistemas, ¿cuál es el archivo señalado por `/sbin/init`?

   El binario principal de systemd: `/lib/systemd/systemd`.

3. ¿Cómo se puede verificar el objetivo predeterminado del sistema (system_target) en un sistema basado en systemd?

   El enlace simbólico `/etc/systemd/system/default.target` apuntará al archivo de unidad definido como el objetivo predeterminado. También se puede usar el comando `systemctl get-default`.

4. ¿Cómo se puede cancelar un reinicio del sistema programado con el comando `shutdown`?

   Se debe usar el comando `shutdown -c`.



Tema 102: Instalación de Linux y gestión de paquetes



102.1 Diseño del esquema de particionado del disco duro duro

Referencia al objetivo del LPI

LPIC-1 v5, Exam 101, Objective 102.1

Importancia

2

Áreas de conocimiento clave

- Asignar sistemas de archivos y espacio de intercambio a particiones o discos separados.
- Adaptar el diseño al uso previsto del sistema.
- Asegurar de que la partición /boot cumple los requisitos de la arquitectura de hardware para el arranque.
- Conocimiento de las características básicas de LVM.

Lista parcial de archivos, términos y utilidades

- Sistema de archivos / (raíz)
- Sistema de archivos `/var`
- Sistema de archivos `/home`
- Sistema de archivos `/boot`
- Partición del sistema EFI (ESP)
- Espacio de intercambio (swap)
- Puntos de montaje
- Particiones



102.1 Lección 1

Certificación:	LPIC-1
Versión:	5.0
Tema:	102 Instalación de Linux y Administración de Paquetes
Objetivo:	102.1 Diseño de Disco Duro
Lección:	1 de 1

Introducción

Para tener éxito en este objetivo, debe comprender la relación entre discos, particiones, sistemas de archivos y volúmenes (*disks*, *partitions*, *filesystems* y *volumes*).

Piense en un disco (o dispositivo de almacenamiento, ya que los dispositivos modernos no contienen ningún "disco" en absoluto) como un "contenedor físico" para sus datos.

Antes de que un disco pueda ser usado por una computadora, necesita ser particionado. Una partición es un subconjunto lógico del disco físico, como una "cerca" lógica. Particionar es una forma de "compartimentar" la información almacenada en el disco, separando, por ejemplo, los datos del sistema operativo de los datos del usuario.

Cada disco necesita al menos una partición, pero puede tener varias particiones si es necesario, y la información en ellas se almacena en una tabla de particiones. Esta tabla incluye información sobre el primer y último sector de la partición y su tipo, así como más detalles sobre cada partición.

Dentro de cada partición hay un sistema de archivos. El sistema de archivos describe la forma en que la información se almacena realmente en el disco. Esta información incluye cómo están organizados

los directorios, cuál es la relación entre ellos, dónde están los datos para cada archivo, etc.

Las particiones no pueden abarcar varios discos. Pero al usar el *Logical Volume Manager* (LVM) se pueden combinar varias particiones, incluso a través de discos, para formar un único volumen lógico.

Los volúmenes lógicos eliminan las limitaciones de los dispositivos físicos y le permiten trabajar con "grupos" de espacio en disco que se pueden combinar o distribuir de una manera mucho más flexible que las particiones tradicionales. LVM es útil en situaciones en las que necesitaría agregar más espacio a una partición sin tener que migrar los datos a un dispositivo más grande.

En este objetivo, aprenderá cómo diseñar un esquema de partición de disco para un sistema Linux, asignando sistemas de archivos e intercambiando espacio para separar particiones o discos cuando sea necesario.

En otras lecciones se discutirá cómo crear y administrar particiones y sistemas de archivos. Discutiremos una visión general de LVM en este objetivo, pero una explicación más detallada está fuera del alcance de esta lección.

Puntos de Montaje

Para poder acceder a un sistema de archivos en Linux, debe estar *montado*. Esto significa adjuntar el sistema de archivos a un punto específico en el árbol de directorios de su sistema, llamado punto de montaje (*mount point*).

Cuando esté montado, el contenido del sistema de archivos estará disponible en el punto de montaje. Por ejemplo, imagine que tiene una partición con los datos personales de sus usuarios (sus directorios de inicio), que contiene los directorios `/john`, `/jack` y `/carol`. Cuando se monta en `/home`, el contenido de esos directorios estará disponible en `/home/john`, `/home/jack` y `/home/carol`.

El punto de montaje debe existir antes de montar el sistema de archivos. No puede montar una partición en `/mnt/userdata` si este directorio no existe. Sin embargo, si el directorio existe y contiene archivos, esos archivos no estarán disponibles hasta que desmonte el sistema de archivos. Si lista los contenidos del directorio, verá los archivos almacenados en el sistema de archivos montado, no los contenidos originales del directorio.

Los sistemas de archivos se pueden montar en cualquier lugar que desee. Sin embargo, hay algunas buenas prácticas que deben seguirse para facilitar la administración del sistema.

Tradicionalmente, `/mnt` era el directorio en el que se montaban todos los dispositivos externos y una cantidad de *puntos de anclaje* preconfigurados para dispositivos comunes, como unidades de CD-ROM (`/mnt/cdrom`) y disquetes (`/mnt/floppy`).

Esto ha sido reemplazado por `/media`, que ahora es el punto de montaje predeterminado para cualquier medio extraíble por el usuario (por ejemplo, discos externos, unidades flash USB, lectores de tarjetas de memoria, discos ópticos, etc.) conectados al sistema.

En la mayoría de las distribuciones modernas de Linux y entornos de escritorio, los dispositivos extraíbles se montan automáticamente en `/media/USER/LABEL` cuando se conectan al sistema, donde `USER` es el nombre de usuario y `LABEL` es la etiqueta del dispositivo. Por ejemplo, una unidad flash USB con la etiqueta `FlashDrive` conectada por el usuario `john` se montaría en `/media/john/FlashDrive/`. La forma en que se maneja esto es diferente según el entorno de escritorio.

Dicho esto, cada vez que necesite *manualmente* montar un sistema de archivos, es una buena práctica montarlo en `/mnt`. Los comandos específicos para controlar el montaje y desmontaje de sistemas de archivos en Linux se analizarán en otra lección.

Manteniendo las cosas separadas

En Linux, hay algunos directorios que deberían mantenerse en particiones separadas. Hay muchas razones para esto: por ejemplo, al mantener los archivos relacionados con el gestor de arranque (almacenados en `/boot`) en una partición de arranque, se asegura de que su sistema aún pueda arrancar en caso de un bloqueo en el sistema de archivos raíz.

Mantener los directorios personales del usuario (en `/home`) en una partición separada facilita la reinstalación del sistema sin el riesgo de tocar accidentalmente los datos del usuario. Mantener los datos relacionados con un servidor web o de base de datos (generalmente en `/var`) en una partición separada (o incluso en un disco separado) facilita la administración del sistema si necesita agregar más espacio en disco para esos casos de uso.

Incluso puede haber razones de rendimiento para mantener ciertos directorios en particiones separadas. Es posible que desee mantener el sistema de archivos raíz (`/`) en una unidad SSD rápida y directorios más grandes como `/home` y `/var` en discos duros más lentos que ofrecen mucho más espacio por una fracción del costo.

La Partición de Arranque (`/boot`)

La partición de arranque contiene archivos utilizados por el gestor de arranque para cargar el sistema operativo. En sistemas Linux, el gestor de arranque suele ser GRUB2 o, en sistemas más antiguos, GRUB Legacy. La partición generalmente se monta en `/boot` y sus archivos se almacenan en `/boot/grub`.

Técnicamente, no se necesita una partición de arranque, ya que en la mayoría de los casos GRUB puede montar la partición raíz (`/`) y cargar los archivos desde un directorio separado `/boot`.

Sin embargo, puede desear una partición de arranque separada por seguridad (garantizando que el sistema se inicie incluso en caso de un bloqueo del sistema de archivos raíz), o si desea utilizar un sistema de archivos que el gestor de arranque no puede entender en la partición raíz, o si utiliza un método de cifrado o compresión no compatible.

La partición de arranque suele ser la primera partición del disco. Esto se debe a que el BIOS original de la PC de IBM definió los discos usando cilindros, cabezas y sectores (*cylinders*, *heads* y *sectors* (CHS)), con un máximo de 1024 cilindros, 256 cabezas y 63 sectores, lo que resulta en un tamaño de disco máximo de 528 MB (504 MB en MS-DOS) . Esto significa que nada más allá de esta marca no sería accesible en los sistemas heredados, a menos que se usara un esquema de direccionamiento de disco diferente (como *Logical Block Addressing*, LBA).

Por lo tanto, para una máxima compatibilidad, la partición de arranque generalmente se encuentra al comienzo del disco y termina antes del cilindro 1024 (528 MB), asegurando que pase lo que pase, la máquina siempre podrá cargar el núcleo.

Dado que la partición de arranque solo almacena los archivos que necesita el gestor de arranque, el disco RAM inicial y las imágenes del núcleo, puede ser bastante pequeño para los estándares actuales. Un buen tamaño es de alrededor de 300 MB.

La partición del sistema EFI (ESP)

La *EFI System Partition* (ESP) es utilizada por máquinas basadas en la interfaz de firmware extensible unificada (*Unified Extensible Firmware Interface* (UEFI)) para almacenar cargadores de arranque e imágenes del núcleo de los sistemas operativos instalados.

Esta partición está formateada en un sistema de archivos basado en FAT. En un disco particionado con una tabla de particiones GUID, tiene un identificador único global `C12A7328-F81F-11D2-BA4B-00A0C93EC93B`. Si el disco fue formateado bajo el esquema de partición MBR, la ID de la partición es `0xEF`.

En las máquinas que ejecutan Microsoft Windows, esta partición suele ser la primera en el disco, aunque esto no es obligatorio. El sistema operativo crea (o completa) el ESP después de la instalación, y en un sistema Linux se monta en `/boot/efi`.

La Partición `/home`

Cada usuario en el sistema tiene un directorio de inicio para almacenar archivos personales y preferencias, y la mayoría de ellos se encuentran en `/home`. Por lo general, el directorio de inicio es el mismo que el nombre de usuario, por lo que el usuario John tendría su directorio en `/home/john`.

Sin embargo, hay excepciones. Por ejemplo, el directorio de inicio para el usuario raíz es `/root` y

algunos servicios del sistema pueden tener usuarios asociados con directorios de inicio en otros lugares.

No existe una regla para determinar el tamaño de una partición para el directorio `/home` (la partición de inicio). Debe tener en cuenta la cantidad de usuarios en el sistema y cómo se utilizará. Un usuario que solo navega por la web y procesa textos requerirá menos espacio que uno que trabaje con la edición de video, por ejemplo.

Información Variable (`/var`)

Este directorio contiene "datos variables", o archivos y directorios en los que el sistema debe poder escribir durante la operación. Esto incluye registros del sistema (en `/var/log`), archivos temporales (`/var/tmp`) y datos de aplicaciones en caché (en `/var/cache`).

`/var/www/html` también es el directorio predeterminado para los archivos de datos del servidor web Apache y `/var/lib/mysql` es la ubicación predeterminada para los archivos de base de datos del servidor MySQL. Sin embargo, ambos pueden ser cambiados.

Una buena razón para poner `/var` en una partición separada es la estabilidad. Muchas aplicaciones y procesos escriben en `/var` y subdirectorios, como `/var/log` o `/var/tmp`. Un proceso con un comportamiento anormal puede escribir datos hasta que no quede espacio libre en el sistema de archivos.

Si `/var` está en `/` esto puede desencadenar en un estado de emergencia del núcleo del sistema operativo (Kernel Panic) y corrupción del sistema de archivos, causando una situación de la que es difícil recuperarse. Pero si `/var` se mantiene en una partición separada, el sistema de archivos raíz no se verá afectado.

Al igual que en `/home`, no existe una regla universal para determinar el tamaño de una partición para `/var`, ya que variará con la forma en que se utiliza el sistema. En un sistema doméstico, puede tomar solo unos pocos gigabytes. Pero en una base de datos o servidor web se puede necesitar mucho más espacio. En tales escenarios, puede ser conveniente colocar `/var` en una partición en un disco diferente al de la partición raíz, agregando una capa adicional de protección contra fallas en el disco físico.

Partición de Intercambio (Swap)

La partición de intercambio se utiliza para intercambiar páginas de memoria de RAM a disco según sea necesario. Esta partición debe ser de un tipo específico y configurarse con una utilidad adecuada llamada `mkswap` antes de poder usarse.

La partición de intercambio no se puede montar como las demás, lo que significa que no puede

acceder a ella como un directorio normal y echar un vistazo a su contenido.

Un sistema puede tener múltiples particiones de intercambio (aunque esto es poco común) y Linux también admite el uso de archivos de intercambio en lugar de particiones, lo que puede ser útil para aumentar rápidamente el espacio de intercambio cuando sea necesario.

El tamaño de la partición de intercambio es un tema polémico. Es posible que la antigua regla de los primeros días de Linux ("dos veces la cantidad de RAM") ya no se aplique dependiendo de cómo se esté utilizando el sistema y la cantidad de RAM física instalada.

En la documentación para Red Hat Enterprise Linux 7, Red Hat recomienda lo siguiente:

Cantidad de RAM	Tamaño de intercambio recomendado	Tamaño de intercambio recomendado con hibernación
< 2 GB of RAM	2x cantidad de RAM	3x cantidad de RAM
2-8 GB of RAM	mismo tamaño RAM	2x cantidad de RAM
8-64 GB of RAM	Al menos 4 GB	1.5x cantidad de RAM
> 64 GB of RAM	Al menos 4 GB	No Recomendado

Por supuesto, la cantidad de intercambio puede depender de la carga de trabajo. Si la máquina está ejecutando un servicio crítico, como una base de datos, un servidor web o SAP, es aconsejable consultar la documentación de estos servicios (o su proveedor de software) para obtener una recomendación.

NOTE Para obtener más información sobre cómo crear y habilitar particiones de intercambio y archivos de intercambio, consulte el Objetivo 104.1 de LPIC-1.

LVM

Ya hemos discutido cómo se organizan los discos en una o más particiones, y cada partición contiene un sistema de archivos que describe cómo se almacenan los archivos y los metadatos asociados. Una de las desventajas de la partición es que el administrador del sistema tiene que decidir de antemano cómo se distribuirá el espacio disponible en un dispositivo de almacenamiento. Esto puede presentar algunos desafíos más adelante, si una partición requiere más espacio de lo planeado originalmente. Por supuesto, las particiones pueden redimensionarse, pero esto puede no ser posible si, por ejemplo, no hay espacio libre en el disco.

Logical Volume Management (LVM) es una forma de virtualización de almacenamiento que ofrece a

los administradores de sistemas un enfoque más flexible para administrar el espacio en disco que la partición tradicional. El objetivo de LVM es facilitar la gestión de las necesidades de almacenamiento de sus usuarios finales. La unidad básica es el *Physical Volume* (PV), que es un dispositivo de bloque en su sistema como una partición de disco o un arreglo RAID.

Los PV se agrupan en *Grupos de volúmenes* (VG) que abstraen los dispositivos subyacentes y se ven como un único dispositivo lógico, con la capacidad de almacenamiento combinada de los componentes del PV.

Cada volumen en un grupo de volúmenes se subdivide en partes de tamaño fijo llamadas *extents*. Las extensiones en un PV se denominan *Physical Extents* (PE), mientras que las de un volumen lógico son *Logical Extents* (LE). En general, cada extensión lógica se asigna a una extensión física, pero esto puede cambiar si se utilizan características como la duplicación de disco.

Los grupos de volúmenes se pueden subdividir en volúmenes lógicos (LV), que funcionan de forma similar a las particiones pero con más flexibilidad.

El tamaño de un volumen lógico, tal como se especificó durante su creación, está definido por el tamaño de las extensiones físicas (4 MB por defecto) multiplicado por el número de extensiones en el volumen. A partir de esto, es fácil comprender que para aumentar un Volumen lógico, por ejemplo, todo lo que el administrador del sistema tiene que hacer es agregar más extensiones del grupo disponible en el Grupo de volúmenes. Del mismo modo, se pueden eliminar extensiones para reducir el LV.

Después de crear un volumen lógico, el sistema operativo lo ve como un dispositivo de bloque normal. Se creará un dispositivo en `/dev`, nombrado como `/dev/VGNAME/LVNAME`, donde `VGNAME` es el nombre del grupo de volúmenes y `LVNAME` es el nombre del volumen lógico.

Estos dispositivos pueden formatearse con el sistema de archivos deseado utilizando utilidades estándares (como `mkfs.ext4`, por ejemplo) y montarse usando los métodos habituales, ya sea manualmente con el comando `mount` o automáticamente agregándolos al archivo `/etc/fstab`.

Ejercicios Guiados

1. En sistemas Linux, ¿dónde se almacenan los archivos para el gestor de arranque GRUB?

2. ¿Dónde debe terminar la partición de arranque para garantizar que una PC siempre pueda cargar el kernel?

3. ¿Dónde se suele montar la partición EFI?

4. Al montar manualmente un sistema de archivos, ¿en qué directorio se debe montar normalmente?

Ejercicios Exploratorios

1. ¿Cuál es la unidad más pequeña dentro de un grupo de volúmenes?

2. ¿Cómo se define el tamaño de un volumen lógico?

3. En un disco formateado con el esquema de partición MBR, ¿cuál es el ID de la partición del sistema EFI?

4. Además de las particiones de intercambio, ¿cómo puede aumentar rápidamente el espacio de intercambio en un sistema Linux?

Resumen

En esta lección aprendió sobre el particionado de discos, qué directorios generalmente se mantienen en particiones separadas y por qué se hace esto. Además, discutimos una descripción general de LVM (Logical Volume Management) y cómo puede ofrecer una forma más flexible de asignar sus datos y espacio en disco en comparación con el particionado tradicional.

Se han discutido los siguientes términos y utilidades:

`/`
: El sistema de archivos raíz de Linux.

`/var`
: La ubicación estándar para "datos variables" , datos que pueden reducirse y crecer con el tiempo.

`/home`
: El directorio padre estándar para directorios de inicio de usuarios regulares en un sistema.

`/boot`
: La ubicación estándar para los archivos del cargador de arranque, el kernel de Linux y el disco RAM inicial.

Partición del sistema EFI (ESP)
: Utilizado por sistemas que tienen UEFI implementado para el almacenamiento de los archivos de arranque del sistema.

Espacio de intercambio
: Se utiliza para intercambiar páginas de memoria del núcleo cuando hay un uso intensivo de la RAM.

Puntos de montaje
: Ubicaciones de directorios donde se montará un dispositivo (como un disco duro).

Particiones
: Divisiones en un disco duro.

Respuestas a los ejercicios guiados

1. En sistemas Linux, ¿dónde se almacenan los archivos para el gestor de arranque GRUB?

   En `/boot/grub`.

2. ¿Dónde debe terminar la partición de arranque para garantizar que una PC siempre pueda cargar el kernel?

   Antes del cilindro 1024.

3. ¿Dónde se suele montar la partición EFI?

   En `/boot/efi`.

4. Al montar manualmente un sistema de archivos, ¿en qué directorio se debe montar normalmente?

   En `/mnt`. Sin embargo, esto no es obligatorio. Puede montar una partición en cualquier directorio que desee.

Respuestas a ejercicios exploratorios

1. ¿Cuál es la unidad más pequeña dentro de un grupo de volúmenes?

   Los grupos de volúmenes se subdividen en extensiones (extends).

2. ¿Cómo se define el tamaño de un volumen lógico?

   Por el tamaño de las extensiones físicas multiplicado por el número de extensiones en el volumen.

3. En un disco formateado con el esquema de partición MBR, ¿cuál es la ID de la partición del sistema EFI?

   El ID es `0xEF`.

4. Además de las particiones de intercambio, ¿cómo puede aumentar rápidamente el espacio de intercambio en un sistema Linux?

   Se pueden usar archivos de intercambio.



102.2 Instalar un gestor de arranque

Referencia al objetivo del LPI

LPIC-1 v5, Exam 101, Objective 102.2

Importancia

2

Áreas de conocimiento clave

- Proporcionar ubicaciones alternativas para el gestor de arranque así como opciones de arranque de respaldo.
- Instalar y configurar un gestor de arranque como GRUB Legacy.
- Realizar cambios básicos de configuración para GRUB 2.
- Interactuar con el gestor de arranque.

Lista parcial de archivos, términos y utilidades

- `menu.lst`, `grub.cfg` y `grub.conf`
- `grub-install`
- `grub-mkconfig`
- MBR



102.2 Lección 1

Certificación:	LPIC-1
Versión:	5.0
Tema:	102 Instalación de Linux y Administración de Paquetes
Objetivo:	102.2 Instalar un administrador de arranque
Lección:	1 de 1

Introducción

Cuando una computadora se enciende, el primer software que se ejecuta es el cargador de arranque. Este es un código cuyo único propósito es cargar el núcleo del sistema operativo y entregarle el control. El núcleo cargará los controladores necesarios, inicializará el hardware y luego cargará el resto del sistema operativo.

GRUB es el gestor de arranque utilizado en la mayoría de las distribuciones de Linux. Puede cargar el núcleo de Linux u otros sistemas operativos, como Windows, y puede manejar múltiples imágenes y parámetros del núcleo del sistema operativo como entradas de menú separadas. La selección del núcleo en el arranque se realiza a través de una interfaz controlada por teclado, y hay una interfaz de línea de comandos para editar las opciones y parámetros de arranque.

La mayoría de las distribuciones de Linux instalan y configuran GRUB (en realidad, GRUB 2) automáticamente, por lo que un usuario normal no necesita pensar en eso. Sin embargo, como administrador del sistema, es vital saber cómo controlar el proceso de arranque para poder recuperar el sistema de una falla de arranque después de una actualización fallida del núcleo, por ejemplo.

En esta lección aprenderá cómo instalar, configurar e interactuar con GRUB.

GRUB Legacy vs. GRUB 2

La versión original de GRUB (*Grand Unified Bootloader*), ahora conocida como *GRUB Legacy* se desarrolló en 1995 como parte del proyecto GNU Hurd, y más tarde se adoptó como el gestor de arranque predeterminado de muchas distribuciones de Linux, reemplazando alternativas anteriores como LILO.

GRUB 2 es una reescritura completa de GRUB con el objetivo de ser más limpio, más seguro, más robusto y más potente. Entre las muchas ventajas sobre GRUB Legacy se encuentran un archivo de configuración mucho más flexible (con muchos más comandos y sentencias condicionales, similar a un lenguaje de script), un diseño más modular y una mejor localización/internacionalización.

También hay soporte para temas y menús gráficos de arranque con pantallas de presentación, la capacidad de arrancar archivos ISO de LiveCD directamente desde el disco duro, mejor soporte para arquitecturas que no son x86, soporte universal para UUID (lo que facilita la identificación de discos y particiones) y mucho más.

GRUB Legacy ya no está en desarrollo activo (la última versión fue 0.97, en 2005), y hoy la mayoría de las principales distribuciones de Linux instalan GRUB 2 como el gestor de arranque predeterminado. Sin embargo, aún puede encontrar sistemas que utilicen GRUB Legacy, por lo que es importante saber cómo usarlo y dónde es diferente de GRUB 2.

¿Dónde se ubica el cargador de arranque?

Históricamente, los discos duros en los sistemas compatibles con PC de IBM se particionaron utilizando el esquema de partición MBR, creado en 1982 para IBM PC-DOS (MS-DOS) 2.0.

En este esquema, el primer sector de 512 bytes del disco se llama *Master Boot Record* y contiene una tabla que describe las particiones en el disco (la tabla de particiones) y también el código de arranque, llamado cargador de arranque.

Cuando se enciende la computadora, este código de gestor de arranque mínimo (debido a restricciones de tamaño) se carga, ejecuta y pasa el control a un cargador de arranque secundario en el disco, generalmente ubicado en un espacio de 32 KB entre el MBR y la primera partición, que cargará los sistemas operativos.

En un disco con particiones MBR, el código de arranque para GRUB está instalado en el MBR. Esto carga y pasa el control a una imagen “núcleo” instalada entre el MBR y la primera partición. Desde este punto, GRUB es capaz de cargar el resto de los recursos necesarios (definiciones de menú, archivos de configuración y módulos adicionales) desde el disco.

Sin embargo, MBR tiene limitaciones en el número de particiones (originalmente un máximo de 4

particiones primarias, luego un máximo de 3 particiones primarias con 1 partición extendida subdividida en un número de particiones lógicas) y tamaños de disco máximos de 2 TB. Para superar estas limitaciones, se creó un nuevo esquema de particionamiento llamado GPT (*GUID Partition Table*), parte del estándar UEFI (*Unified Extensible Firmware Interface*).

Los discos con particiones GPT se pueden usar con computadoras con el BIOS de PC tradicional o con el firmware UEFI. En máquinas con un BIOS, la segunda parte de GRUB se almacena en una partición especial de arranque del BIOS.

En los sistemas con firmware UEFI, GRUB se carga mediante el firmware desde los archivos `grubia32.efi` (para sistemas de 32 bits) o `grubx64.efi` (para sistemas de 64 bits) desde una partición llamada ESP (*EFI System Partition*).

La partición `/boot`

En Linux, los archivos necesarios para el proceso de arranque generalmente se almacenan en una partición de arranque, montados en el sistema de archivos raíz y coloquialmente denominados `/boot`.

No se necesita una partición de arranque en los sistemas actuales, ya que los cargadores de arranque como GRUB generalmente pueden montar el sistema de archivos raíz y buscar los archivos necesarios dentro de un directorio `/boot`, pero es una buena práctica ya que separa los archivos necesarios para proceso de arranque desde el resto del sistema de archivos.

Esta partición suele ser la primera en el disco. Esto se debe a que el BIOS original de la PC de IBM diseñó los discos usando Cilindros, Cabezas y Sectores (*Cylinders*, *Heads* y *Sectors* (CHS)), con un máximo de 1024 cilindros, 256 cabezas y 63 sectores, lo que resulta en un tamaño de disco máximo de 528 MB (504 MB sobre MS-DOS) . Esto significa que nada más allá de esta marca no sería accesible, a menos que se utilizara un esquema de direccionamiento de disco diferente (como LBA, *Logical Block Addressing*).

Entonces, para una máxima compatibilidad, la partición `/boot` generalmente se encuentra al comienzo del disco y termina antes del cilindro 1024 (528 MB), asegurando que la máquina siempre pueda cargar el kernel. El tamaño recomendado para esta partición en una máquina actual es de 300 MB.

Otras razones para una partición `/boot` separada son el cifrado y la compresión, ya que algunos métodos pueden no ser compatibles con GRUB 2 todavía, o si necesita tener la partición raíz del sistema (`/`) formateada utilizando un sistema de archivos no compatible.

Contenido de la partición de arranque

El contenido de la partición `/boot` puede variar con la arquitectura del sistema o el cargador de arranque en uso, pero en un sistema basado en x86, generalmente encontrará los archivos a continuación. La mayoría de estos se nombran con un sufijo `-VERSION`, donde `-VERSION` es la versión del núcleo de Linux correspondiente. Entonces, por ejemplo, un archivo de configuración para la versión del núcleo de Linux `4.15.0-65-generic` se llamaría `config-4.15.0-65-generic`.

Archivo de configuración

Este archivo, generalmente llamado `config-VERSION` (vea el ejemplo anterior), almacena los parámetros de configuración para el núcleo de Linux. Este archivo se genera automáticamente cuando se compila o instala un nuevo núcleo y el usuario no debe modificarlo directamente.

Mapa del sistema

Este archivo es una tabla de búsqueda que combina nombres de símbolos (como variables o funciones) con su posición correspondiente en la memoria. Esto es útil al depurar un tipo de falla del sistema conocida como *kernel panic*, ya que permite al usuario saber qué variable o función se estaba llamando cuando ocurrió la falla. Al igual que el archivo de configuración, el nombre suele ser `System.map-VERSION` (por ejemplo, `System.map-4.15.0-65-generic`).

Kernel de Linux

Este es el núcleo del sistema operativo propiamente dicho. El nombre suele ser `vmlinux-VERSION` (por ejemplo, `vmlinux-4.15.0-65-generic`). También puede encontrar el nombre `vmlinuz` en lugar de `vmlinux`, la `z` al final significa que el archivo ha sido comprimido.

Disco RAM inicial

Esto generalmente se llama `initrd.img-VERSION` y contiene un sistema de archivos raíz mínimo cargado en un disco RAM, que contiene utilidades y módulos del núcleo necesarios para que el núcleo pueda montar el sistema de archivos raíz real.

Archivos relacionados con el cargador de arranque

En los sistemas con GRUB instalado, estos generalmente se encuentran en `/boot/grub` e incluyen el archivo de configuración GRUB (`/boot/grub/grub.cfg` para GRUB 2 o `/boot/grub/menu.lst` en caso de GRUB Legacy), módulos (en `/boot/grub/i386-pc`), archivos de traducción (en `/boot/grub/locale`) y fuentes (en `/boot/grub/fonts`).

GRUB 2

Instalando GRUB 2

GRUB 2 se puede instalar utilizando la utilidad `grub-install`. Si tiene un sistema que no arranca, necesitará arrancar usando un Live CD o un disco de rescate, averiguar cuál es la partición de arranque de su sistema, montarlo y luego ejecutar la utilidad.

> **NOTE** Los siguientes comandos suponen que ha iniciado sesión como root. Si no, primero ejecute `sudo su -` para “convertirse en” root. Cuando termine, escriba `exit` para cerrar sesión y volver a un usuario normal.

El primer disco de un sistema suele ser el *boot device* y es posible que necesite saber si hay una *boot partition* en el disco. Esto se puede hacer con la utilidad `fdisk`. Para enumerar todas las particiones en el primer disco de su máquina, use:

```
# fdisk -l /dev/sda
Disk /dev/sda: 111,8 GiB, 120034123776 bytes, 234441648 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: dos
Disk identifier: 0x97f8fef5

Device     Boot    Start       End   Sectors   Size Id Type
/dev/sda1  *        2048   2000895   1998848   976M 83 Linux
/dev/sda2        2002942 234440703 232437762 110,9G  5 Extended
/dev/sda5        2002944  18008063  16005120   7,6G 82 Linux swap / Solaris
/dev/sda6       18010112 234440703 216430592 103,2G 83 Linux
```

La partición de arranque se identifica con el `*` debajo de la columna de arranque. En el ejemplo anterior, es `/dev/sda1`.

Ahora, cree un directorio temporal en `/mnt` y monte la partición en él:

```
# mkdir /mnt/tmp
# mount /dev/sda1 /mnt/tmp
```

Luego ejecute `grub-install`, apuntándolo al dispositivo de arranque (*no* la partición) y al directorio donde está montada la partición de arranque. Si su sistema tiene una partición de arranque dedicada, el comando es:

```
# grub-install --boot-directory=/mnt/tmp /dev/sda
```

Si está instalando en un sistema que no tiene una partición de arranque, sino solo un directorio `/boot` en el sistema de archivos raíz, utilice `grub-install`. Entonces, el comando es:

```
# grub-install --boot-directory=/boot /dev/sda
```

Configurando GRUB 2

El archivo de configuración predeterminado para GRUB 2 es `/boot/grub/grub.cfg`. Este archivo se genera automáticamente y no se recomienda la edición manual. Para realizar cambios en la configuración de GRUB, debe editar el archivo `/etc/default/grub` y luego ejecutar la utilidad `update-grub` para generar un archivo compatible.

NOTE `update-grub` suele ser un acceso directo a `grub-mkconfig -o /boot/grub/grub.cfg`, por lo que producen los mismos resultados.

Hay algunas opciones en el archivo `/etc/default/grub` que controlan el comportamiento de GRUB 2, como el kernel predeterminado para arrancar, el tiempo de espera, los parámetros adicionales de la línea de comandos, etc. Los más importantes son:

`GRUB_DEFAULT=`

La entrada de menú predeterminada para arrancar. Puede ser un valor numérico (como `0`, `1`, etc.), el nombre de una entrada de menú (como `debian`) o `saved`, que se usa junto con `GRUB_SAVEDEFAULT=`, explicado a continuación. Tenga en cuenta que las entradas del menú comienzan en cero, por lo que la primera entrada del menú es `0`, la segunda es `1`, etc.

`GRUB_SAVEDEFAULT=`

Si esta opción se establece en `true` y `GRUB_DEFAULT=` se establece en `saved`, entonces la opción de inicio predeterminada siempre será la última seleccionada en el menú de inicio.

`GRUB_TIMEOUT=`

El tiempo de espera, en segundos, antes de que se seleccione la entrada de menú predeterminada. Si se establece en `0`, el sistema iniciará la entrada predeterminada sin mostrar un menú. Si se establece en `-1`, el sistema esperará hasta que el usuario seleccione una opción, sin importar cuánto tiempo tarde.

`GRUB_CMDLINE_LINUX=`

Esto enumera las opciones de línea de comando que se agregarán a las entradas para el kernel de Linux.

`GRUB_CMDLINE_LINUX_DEFAULT=`

Por defecto, se generan dos entradas de menú para cada núcleo de Linux, una con las opciones predeterminadas y una entrada para la recuperación. Con esta opción, puede agregar parámetros adicionales que se agregarán solo a la entrada predeterminada.

`GRUB_ENABLE_CRYPTODISK=`

Si se establece en `y`, los comandos como `grub-mkconfig`, `update-grub` y `grub-install` buscarán discos cifrados y agregarán los comandos necesarios para acceder a ellos durante el arranque. Esto desactiva el arranque automático (`GRUB_TIMEOUT=` con cualquier valor que no sea `-1`) porque se necesita una contraseña para descifrar los discos antes de que se pueda acceder a ellos.

Administrar entradas de menú

Cuando se ejecuta `update-grub`, GRUB 2 buscará núcleos y sistemas operativos en la máquina y generarán las entradas de menú correspondientes en el archivo `/boot/grub/grub.cfg`. Se pueden agregar nuevas entradas manualmente a los archivos de script dentro del directorio `/etc/grub.d`.

Estos archivos deben ser ejecutables y son procesados en orden numérico por `update-grub`. Por lo tanto, `05_debian_theme` se procesa antes que `10_linux` y así sucesivamente. Las entradas de menú personalizadas generalmente se agregan al archivo `40_custom`.

La sintaxis básica para una entrada de menú se muestra a continuación:

```
menuentry "Default OS" {
    set root=(hd0,1)
    linux /vmlinuz root=/dev/sda1 ro quiet splash
    initrd /initrd.img
}
```

La primera línea siempre comienza con `menuentry` y termina con `{`. El texto entre comillas se mostrará como la etiqueta de entrada en el menú de arranque de GRUB 2.

El parámetro `set root` define el disco y la partición donde se encuentra el sistema de archivos raíz para el sistema operativo. Tenga en cuenta que en GRUB 2 los discos están numerados desde cero, por lo que `hd0` es el primer disco (`sda` en Linux), `hd1` el segundo, y así sucesivamente. Las particiones, sin embargo, están numeradas a partir de uno. En el ejemplo anterior, el sistema de archivos raíz se

encuentra en el primer disco (`hd0`), la primera partición (`,1`) o `sda1`.

En lugar de especificar directamente el dispositivo y la partición, también puede hacer que GRUB 2 busque un sistema de archivos con una etiqueta específica o UUID (*Universally Unique Identifier*). Para eso, utilice el parámetro `search --set=root` seguido del parámetro `--label` y la etiqueta del sistema de archivos para buscar, o `--fs-uuid` seguido del UUID del sistema de archivos.

Puede encontrar el UUID de un sistema de archivos con el siguiente comando:

```
$ ls -l /dev/disk/by-uuid/
total 0
lrwxrwxrwx 1 root root 10 nov  4 08:40 3e0b34e2-949c-43f2-90b0-25454ac1595d ->
../../sda5
lrwxrwxrwx 1 root root 10 nov  4 08:40 428e35ee-5ad5-4dcb-adca-539aba6c2d84 ->
../../sda6
lrwxrwxrwx 1 root root 10 nov  5 19:10 56C11DCC5D2E1334 -> ../../sdb1
lrwxrwxrwx 1 root root 10 nov  4 08:40 ae71b214-0aec-48e8-80b2-090b6986b625 ->
../../sda1
```

En el ejemplo anterior, el UUID para `/dev/sda1` es `ae71b214-0aec-48e8-80b2-090b6986b625`. Si desea establecerlo como el dispositivo raíz para GRUB 2, el comando sería `search --set=root --fs-uuid ae71b214-0aec-48e8-80b2-090b6986b625`.

Cuando se usa el comando `search`, es común agregar el parámetro `--no-floppy` para que GRUB no pierda el tiempo buscando en disquetes.

La línea `linux` indica dónde se encuentra el núcleo del sistema operativo (en este caso, el archivo `vmlinuz` en la raíz del sistema de archivos). Después de eso, puede pasar los parámetros de la línea de comandos al núcleo del sistema operativo.

En el ejemplo anterior, especificamos la partición raíz (`root=/dev/sda1`) y pasamos tres parámetros del kernel: la partición raíz debe montarse como solo lectura (`ro`), la mayoría de los mensajes de registro deben estar deshabilitados (`quiet`) y se debe mostrar una pantalla de bienvenida (`splash`).

La línea `initrd` indica dónde se encuentra el disco RAM inicial. En el ejemplo anterior, el archivo es `initrd.img`, ubicado en la raíz del sistema de archivos.

NOTE La mayoría de las distribuciones de Linux no colocan el núcleo y el initrd en el directorio raíz del sistema de archivos. En cambio, estos son enlaces a los archivos reales dentro del directorio o partición `/boot`.

La última línea de una entrada de menú debe contener solo el carácter `}`.

Interactuando con GRUB 2

Al iniciar un sistema con GRUB 2, verá un menú de opciones. Use las teclas de flecha para seleccionar una opción y `Enter` para confirmar y arrancar la entrada seleccionada.

> **TIP** Si ve solo una cuenta regresiva, pero no un menú, presione `Shift` para que aparezca el menú.

Para editar una opción, selecciónela con las teclas de flecha y presione `E`. Esto mostrará una ventana del editor con el contenido del `menuentry` asociado con esa opción, como se define en `/boot/grub/grub.cfg`.

Después de editar una opción, escriba `Ctrl` + `X` o `F10` para arrancar, o `Esc` para volver al menú.

Para ingresar al shell de GRUB 2, presione `C` en la pantalla del menú (o `Ctrl` + `C`) en la ventana de edición). Verá un símbolo del sistema como este: `grub>`

Escriba `help` para ver una lista de todos los comandos disponibles, o presione `Esc` para salir del shell y volver a la pantalla del menú.

> **NOTE** Recuerde que este menú no aparecerá si `GRUB_TIMEOUT` está configurado en `0` en `/etc/default/grub`.

Arranque desde la consola del GRUB 2

Puede usar el shell GRUB 2 para arrancar el sistema en caso de que una configuración incorrecta en una entrada del menú haga que falle.

Lo primero que debe hacer es averiguar dónde está la partición de arranque. Puede hacerlo con el comando `ls`, que le mostrará una lista de las particiones y discos que GRUB 2 ha encontrado.

```
grub> ls
(proc) (hd0) (hd0,msdos1)
```

En el ejemplo anterior, las cosas son fáciles. Solo hay un disco `(hd0)` con solo una partición: `(hd0,msdos1)`.

Los discos y particiones enumerados serán diferentes en su sistema. En nuestro ejemplo, la primera

partición de `hd0` se llama `msdos1` porque el disco se particionó utilizando el esquema de partición MBR. Si se particionara usando GPT, el nombre sería `gpt1`.

Para arrancar Linux, necesitamos un kernel y un disco RAM inicial (initrd). Veamos el contenido de `(hd0, msdos1)`:

```
grub> ls (hd0,msdos1)/
lost+found/ swapfile etc/ media/ bin/ boot/ dev/ home/ lib/ lib64/ mnt/ opt/ proc/
root/ run/ sbin/ srv/ sys/ tmp/ usr/ var/ initrd.img initrd.img.old vmlinuz cdrom/
```

Puede agregar el parámetro `-l` a `ls` para obtener una lista larga, similar a lo que obtendría en un terminal Linux. Use *Tab* para autocompletar los nombres de disco, partición y archivo.

Tenga en cuenta que tenemos imágenes de kernel (`vmlinuz`) e initrd (`initrd.img`) directamente en el directorio raíz. Si no, podríamos verificar el contenido de `/boot` con `list (hd0,msdos1)/boot/`.

Ahora, configure la partición de arranque:

```
grub> set root=(hd0,msdos1)
```

Cargue el kernel de Linux con el comando `linux`, seguido de la ruta al kernel y la opción `root=` para decirle al kernel dónde se encuentra el sistema de archivos raíz para el sistema operativo.

```
grub> linux /vmlinuz root=/dev/sda1
```

Cargue el disco RAM inicial con `initrd`, seguido de la ruta completa al archivo `initrd.img`:

```
grub> initrd /initrd.img
```

Ahora, inicie el sistema con `boot`.

Arranque desde la consola de rescate

En caso de una falla de arranque, GRUB 2 puede cargar un shell de rescate, una versión simplificada del shell que mencionamos anteriormente. Lo reconocerá mediante el símbolo del sistema, que se muestra como `grub rescue>`.

El proceso para iniciar un sistema desde esta consola es casi el mismo que se muestra anteriormente.

Sin embargo, deberá cargar algunos módulos GRUB 2 para que todo funcione.

Después de descubrir qué partición es la partición de arranque (con `ls`, como se muestra antes), use el comando `set prefix=`, seguido de la ruta completa al directorio que contiene los archivos GRUB 2. Usualmente `/boot/grub`. En nuestro ejemplo:

```
grub rescue> set prefix=(hd0,msdos1)/boot/grub
```

Ahora, cargue los módulos `normal` y `linux` con el comando `insmod`:

```
grub rescue> insmod normal
grub rescue> insmod linux
```

Luego, configure la partición de arranque con `set root=` como se indicó anteriormente, cargue el kernel de Linux (con `linux`), el disco RAM inicial (`initrd`) e intente arrancar con `boot`.

GRUB Legacy

Instalación de GRUB Legacy desde un sistema en ejecución

Para instalar GRUB Legacy en un disco desde un sistema en ejecución, utilizaremos la utilidad `grub-install`. El comando básico es `grub-install DEVICE` donde `DEVICE` es el disco donde desea instalar GRUB Legacy. Un ejemplo sería `/dev/sda`.

```
# grub-install /dev/sda
```

Tenga en cuenta que debe especificar el *device* donde se instalará GRUB Legacy, como `/dev/sda/`, *no la partición* como en `/dev/sda1`.

Por defecto, GRUB copiará los archivos necesarios al directorio `/boot` en el dispositivo especificado. Si desea copiarlos a otro directorio, use el parámetro `--boot-directory=`, seguido de la ruta completa a donde se deben copiar los archivos.

Instalación de GRUB Legacy desde un GRUB Shell

Si no puede iniciar el sistema por algún motivo y necesita reinstalar GRUB Legacy, puede hacerlo desde la consola de GRUB en un disco de inicio de GRUB Legacy.

Desde el shell de GRUB (escriba `c` en el menú de arranque para acceder al indicador `grub>`), el primer

paso es configurar el dispositivo de arranque, que contiene el directorio `/boot`. Por ejemplo, si este directorio está en la primera partición del primer disco, el comando sería:

```
grub> root (hd0,0)
```

Si no sabe qué dispositivo contiene el directorio `/boot`, puede pedirle a GRUB que lo busque con el comando `find`, como se muestra a continuación:

```
grub> find /boot/grub/stage1
 (hd0,0)
```

Luego, configure la partición de arranque como se indicó anteriormente y use el comando `setup` para instalar GRUB Legacy en el MBR y copie los archivos necesarios en el disco:

```
grub> setup (hd0)
```

Cuando termine, reinicie el sistema y debería arrancar normalmente.

Configuración de entradas y ajustes del menú GRUB Legacy

Las entradas y configuraciones de menú de GRUB Legacy se almacenan en el archivo `/boot/grub/menu.lst`. Este es un archivo de texto simple con una lista de comandos y parámetros, que puede editarse directamente con su editor de texto favorito.

Las líneas que comienzan con `#` se consideran comentarios y las líneas en blanco se ignoran.

Una entrada de menú tiene al menos tres comandos. El primero, `title`, establece el título del sistema operativo en la pantalla del menú. El segundo, `root`, le dice a GRUB legado desde qué dispositivo o partición arrancar.

La tercera entrada, `kernel`, especifica la ruta completa a la imagen del núcleo del sistema operativo que debe cargarse cuando se selecciona la entrada correspondiente. Tenga en cuenta que esta ruta es relativa al dispositivo especificado en el parámetro `root`.

A continuación, un ejemplo simple:

```
# This line is a comment
title My Linux Distribution
root (hd0,0)
kernel /vmlinuz root=/dev/hda1
```

A diferencia de GRUB 2, en GRUB Legacy ambos discos *y* particiones están numerados desde cero. Entonces, el comando `root (hd0,0)` establecerá la partición de arranque como la primera partición (`0`) del primer disco (`hd0`).

Puede omitir la instrucción `root` si especifica el dispositivo de arranque antes de la ruta en el comando `kernel`. La sintaxis es la misma, entonces:

```
kernel (hd0,0)/vmlinuz root=dev/hda1
```

es equivalente a:

```
root (hd0,0)
kernel /vmlinuz root=/dev/hda1
```

Ambos cargarán el archivo `vmlinuz` desde el directorio raíz (/) de la primera partición del primer disco (`hd0,0`).

El parámetro `root=/dev/hda1` después del comando `kernel` le dice al kernel de Linux qué partición debe usarse como sistema de archivos raíz. Este es un parámetro del núcleo de Linux, no un comando GRUB legacy.

> **NOTE** Para obtener más información sobre los parámetros del núcleo de Linux, visite https://www.kernel.org/doc/html/v4.14/admin-guide/kernel-parameters.html.

Es posible que deba especificar la ubicación de la imagen de disco RAM inicial para el sistema operativo con el parámetro `initrd`. La ruta completa al archivo puede especificarse como en el parámetro `kernel`, y también puede especificar un dispositivo o partición antes de la ruta, por ejemplo:

```
# This line is a comment
title My Linux Distribution
root (hd0,0)
kernel /vmlinuz root=/dev/hda1
initrd /initrd.img
```

GRUB Legacy tiene un diseño modular, donde los módulos (generalmente almacenados como archivos `.mod` en `/boot/grub/i386-pc`) se pueden cargar para agregar funciones adicionales, como soporte para hardware inusual, sistemas de archivos o nuevos algoritmos de compresión.

Los módulos se cargan utilizando el comando `module`, seguido de la ruta completa al archivo `.mod` correspondiente. Tenga en cuenta que, al igual que los núcleos y las imágenes initrd, esta ruta es relativa al dispositivo especificado en el comando `root`.

El siguiente ejemplo cargará el módulo `915resolution`, necesario para establecer correctamente la resolución de framebuffer en sistemas con conjuntos de chips de video Intel de las series 800 o 900.

```
module /boot/grub/i386-pc/915resolution.mod
```

Carga en cadena de otros sistemas operativos

GRUB Legacy se puede usar para cargar sistemas operativos no compatibles, como Windows, mediante un proceso llamado *chainloading*. GRUB Legacy se carga primero, y cuando se selecciona la opción correspondiente, se carga el gestor de arranque para el sistema deseado.

Una entrada típica para cargar Windows en cadena se vería como la siguiente:

```
# Load Windows
title Windows XP
root (hd0,1)
makeactive
chainload +1
boot
```

Pasemos por cada parámetro. Como antes, `root (hd0,1)` especifica el dispositivo y la partición donde se encuentra el cargador de arranque para el sistema operativo que deseamos cargar. En este ejemplo, la *segunda* partición del primer disco.

`makeactive`
: establecerá una bandera que indica que esta es una partición activa. Esto solo funciona en particiones primarias de DOS.

`chainload +1`
: le dice a GRUB que cargue el primer sector de la partición de arranque. Aquí es donde generalmente se encuentran los gestores de arranque.

`boot`
: ejecutará el gestor de arranque y cargará el sistema operativo correspondiente.

Ejercicios Guiados

1. ¿Cuál es la ubicación predeterminada para el archivo de configuración GRUB 2?

2. ¿Cuáles son los pasos necesarios para cambiar la configuración de GRUB 2?

3. ¿En qué archivo se deben agregar entradas de menú personalizadas de GRUB 2?

4. ¿Dónde se almacenan las entradas de menú para GRUB Legacy?

5. Desde un menú GRUB 2 o GRUB Legacy, ¿cómo puede ingresar a la consola de GRUB?

Ejercicios Exploratorios

1. Imagine un usuario que configura GRUB Legacy para arrancar desde la segunda partición del primer disco. Escribe la siguiente entrada de menú personalizada:

```
title My Linux Distro
root (hd0,2)
kernel /vmlinuz root=/dev/hda1
initrd /initrd.img
```

   Sin embargo, el sistema no se iniciará. ¿Qué está mal?

2. Imagine que tiene un disco identificado como `/dev/sda` con múltiples particiones. ¿Qué comando se puede usar para averiguar cuál es la partición de arranque en un sistema?

3. ¿Qué comando se puede usar para averiguar el UUID de una partición?

4. Considere la siguiente entrada para GRUB 2

```
menuentry "Default OS" {
    set root=(hd0,1)
    linux /vmlinuz root=/dev/sda1 ro quiet splash
    initrd /initrd.img
}
```

   Cámbiela para que el sistema arranque desde un disco con el UUID `5dda0af3-c995-481a-a6f3-46dcd3b6998d`

5. ¿Cómo puede configurar GRUB 2 para que espere 10 segundos antes de iniciar la entrada de menú predeterminada?

6. Desde una consola GRUB Legacy, ¿cuáles son los comandos para instalar GRUB en la primera partición del segundo disco?

Resumen

En esta lección aprendimos

- ¿Qué es un gestor de arranque?.
- Las diferencias entre GRUB Legacy y GRUB 2.
- ¿Qué es una partición de arranque y cuáles son sus contenidos?
- ¿Cómo instalar GRUB Legacy y GRUB 2?
- ¿Cómo configurar GRUB Legacy y GRUB 2?
- ¿Cómo agregar entradas de menú personalizadas a GRUB Legacy y GRUB 2?
- ¿Cómo interactuar con la pantalla del menú y la consola de GRUB Legacy y GRUB 2?
- ¿Cómo arrancar un sistema desde un shell GRUB Legacy o GRUB 2 o una consola de rescate?

Los siguientes comandos se discutieron en esta lección:

- `grub-install`
- `update-grub`
- `grub-mkconfig`

Respuestas a los ejercicios guiados

1. ¿Cuál es la ubicación predeterminada para el archivo de configuración GRUB 2?

   `/boot/grub/grub.cfg`

2. ¿Cuáles son los pasos necesarios para cambiar la configuración de GRUB 2?

   Realice los cambios en el archivo `/etc/default/grub`, luego actualice la configuración con `update-grub`.

3. ¿En qué archivo se deben agregar entradas de menú personalizadas de GRUB 2?

   `/etc/grub.d/40_custom`

4. ¿Dónde se almacenan las entradas de menú para GRUB Legacy?

   `/boot/grub/menu.lst`

5. Desde un menú GRUB 2 o GRUB Legacy, ¿cómo puede ingresar a la consola de GRUB?

   Presione `c` en la pantalla del menú.

Respuestas a ejercicios exploratorios

1. Imagine un usuario que configura GRUB Legado para arrancar desde la segunda partición del primer disco. Escribe la siguiente entrada de menú personalizada:

```
title My Linux Distro
root (hd0,2)
kernel /vmlinuz root=/dev/hda1
initrd /initrd.img
```

   Sin embargo, el sistema no se iniciará. ¿Qué está mal?

   La partición de arranque está mal. Recuerde que, a diferencia de GRUB 2, GRUB Legacy cuenta las particiones a partir de cero (*zero*). Entonces, el comando correcto para la segunda partición del primer disco debe ser `root (hd0,1)`.

2. Imagine que tiene un disco identificado como `/dev/sda` con múltiples particiones. ¿Qué comando se puede usar para averiguar cuál es la partición de arranque en un sistema?

   Use `fdisk -l /dev/sda`. La partición de arranque se marcará con un asterisco (*) en la lista.

3. ¿Qué comando se puede usar para averiguar el UUID de una partición?

   Use `ls -la /dev/disk/by-uuid/` y busque el UUID que apunta a la partición.

4. Considere la siguiente entrada para GRUB 2

```
menuentry "Default OS" {
    set root=(hd0,1)
    linux /vmlinuz root=/dev/sda1 ro quiet splash
    initrd /initrd.img
}
```

   Cámbiela para que el sistema arranque desde un disco con el UUID `5dda0af3-c995-481a-a6f3-46dcd3b6998d`

   Deberá cambiar la instrucción `set root`. En lugar de especificar un disco y una partición, configure el grub de manera que busque la partición con el UUID deseado.

```
menuentry "Default OS" {
    search --set=root --fs-uuid 5dda0af3-c995-481a-a6f3-46dcd3b6998d
    linux /vmlinuz root=/dev/sda1 ro quiet splash
    initrd /initrd.img
}
```

5. ¿Cómo puede configurar GRUB 2 para que espere 10 segundos antes de iniciar la entrada de menú predeterminada?

   Agregue el parámetro `GRUB_TIMEOUT=10` a `/etc/default/grub`.

6. Desde una consola GRUB Legacy, ¿cuáles son los comandos para instalar GRUB en la primera partición del segundo disco?

```
grub> root (hd1,0)
grub> setup (hd1)
```



102.3 Gestión de librerías compartidas

Referencia al objetivo del LPI

LPIC-1 v5, Exam 101, Objective 102.3

Importancia

1

Áreas de conocimiento clave

- Identificar librerías compartidas.
- Identificar las ubicaciones típicas de las librerías del sistema.
- Cargar librerías compartidas.

Lista parcial de archivos, términos y utilidades

- `ldd`
- `ldconfig`
- `/etc/ld.so.conf`
- `LD_LIBRARY_PATH`



102.3 Lección 1

Certificación:	LPIC-1
Versión:	5.0
Tema:	102 Instalación de Linux y Administración de Paquetes
Objetivo:	102.3 Administrar bibliotecas compartidas
Lección:	1 de 1

Introducción

En esta lección discutiremos sobre bibliotecas compartidas (*shared libraries*), también conocidas como objetos compartidos (*shared objects*): partes de código compilado y reutilizable como funciones o clases, que varios programas utilizan de manera recurrente.

Para comenzar, explicaremos qué son las bibliotecas compartidas, cómo identificarlas y dónde se encuentran. A continuación, veremos cómo configurar sus ubicaciones de almacenamiento. Finalmente, mostraremos cómo buscar las bibliotecas compartidas de las que depende un programa en particular.

Concepto de bibliotecas compartidas

Al igual que sus contrapartes físicas, las bibliotecas de software son colecciones de código que están destinadas a ser utilizadas por muchos programas diferentes; así como las bibliotecas físicas guardan libros y otros recursos para ser utilizados por muchas personas diferentes.

Para construir un archivo ejecutable a partir del código fuente de un programa, son necesarios dos pasos importantes. Primero, el *compilador* convierte el código fuente en código de máquina que se

almacena en los llamados *object files*. En segundo lugar, el *linker* combina los archivos de objetos y los vincula a las bibliotecas para generar el archivo ejecutable final. Este enlace puede hacerse *statically* (estáticamente) o *dynamically* (dinámicamente). Dependiendo del método que utilicemos, hablaremos de bibliotecas estáticas o, en caso de vinculación dinámica, de bibliotecas compartidas. Expliquemos sus diferencias.

Bibliotecas estáticas

Una biblioteca estática se fusiona con el programa en el momento del enlace. Una copia del código de la biblioteca se incrusta en el programa y se convierte en parte de él. Por lo tanto, el programa no tiene dependencias de la biblioteca en tiempo de ejecución porque el programa ya contiene el código de la biblioteca. No tener dependencias puede verse como una ventaja, ya que no tiene que preocuparse por asegurarse de que las bibliotecas utilizadas siempre estén disponibles. En el lado negativo, los programas vinculados estáticamente son más pesados.

Bibliotecas compartidas (o dinámicas)

En el caso de las bibliotecas compartidas, el enlazador simplemente se encarga de que el programa haga referencia a las bibliotecas correctamente. Sin embargo, el vinculador no copia ningún código de biblioteca en el archivo del programa. Sin embargo, en tiempo de ejecución, la biblioteca compartida debe estar disponible para satisfacer las dependencias del programa. Este es un enfoque económico para administrar los recursos del sistema, ya que ayuda a reducir el tamaño de los archivos de programa y solo se carga una copia de la biblioteca en la memoria, incluso cuando es utilizada por varios programas.

Convenciones de nomenclatura de archivos de objetos compartidos

El nombre de una biblioteca compartida, también conocida como *soname*, sigue un patrón que se compone de tres elementos:

- Nombre de la biblioteca (normalmente precedido por `lib`)
- `so` (que significa “objeto compartido”)
- Número de versión de la biblioteca

Por ejemplo: `libpthread.so.0`

Por el contrario, los nombres de las bibliotecas estáticas terminan en `.a`, p. ej. `libpthread.a`.

NOTE Debido a que los archivos que contienen bibliotecas compartidas deben estar disponibles cuando se ejecuta el programa, la mayoría de los sistemas Linux contienen bibliotecas compartidas. Dado que las bibliotecas estáticas solo se requieren en un archivo dedicado cuando se vincula un programa, es posible que no estén presentes en un sistema de usuario final.

`glibc` (biblioteca GNU C) es un buen ejemplo de una biblioteca compartida. En un sistema Debian GNU/Linux 9.9, su archivo se llama `libc.so.6`. Tales nombres de archivo bastante generales son normalmente enlaces simbólicos que apuntan al archivo real que contiene una biblioteca, cuyo nombre contiene el número de versión exacto. En el caso de `glibc`, este enlace simbólico se ve así:

```
$ ls -l /lib/x86_64-linux-gnu/libc.so.6
lrwxrwxrwx 1 root root 12 feb  6 22:17 /lib/x86_64-linux-gnu/libc.so.6 -> libc-
2.24.so
```

Este patrón de hacer referencia a archivos de biblioteca compartida nombrados por una versión específica por nombres de archivo más generales es una práctica común.

Otros ejemplos de bibliotecas compartidas incluyen `libreadline` (que permite a los usuarios editar líneas de comando a medida que se escriben e incluye soporte para los modos de edición Emacs y vi), `libcrypt` (que contiene funciones relacionadas con el cifrado, el hash y la codificación) , o `libcurl` (que es una biblioteca de transferencia de archivos multiprotocolo).

Las ubicaciones comunes para las bibliotecas compartidas en un sistema Linux son:

- `/lib`
- `/lib32`
- `/lib64`
- `/usr/lib`
- `/usr/local/lib`

NOTE El concepto de bibliotecas compartidas no es exclusivo de Linux. En Windows, por ejemplo, se denominan DLL, que significa *dynamic linked libraries* (bibliotecas vinculadas dinámicamente).

Configuración de rutas de bibliotecas compartidas

Las referencias contenidas en los programas vinculados dinámicamente se resuelven mediante el vinculador dinámico (`ld.so` o `ld-linux.so`) cuando se ejecuta el programa. El vinculador dinámico busca bibliotecas en varios directorios. Estos directorios están especificados por la ruta de la biblioteca. La ruta de la biblioteca se configura en el directorio `/etc`, es decir, en el archivo `/etc/ld.so.conf` y, más común hoy en día, en archivos que residen en el directorio `/etc/ld.so.conf.d`. Normalmente, el primero incluye una sola línea `include` para los archivos `*.conf` en el segundo:

```
$ cat /etc/ld.so.conf
include /etc/ld.so.conf.d/*.conf
```

El directorio `/etc/ld.so.conf.d` contiene archivos `*.conf`:

```
$ ls /etc/ld.so.conf.d/
libc.conf  x86_64-linux-gnu.conf
```

Estos archivos `*.conf` deben incluir las rutas absolutas a los directorios de las bibliotecas compartidas:

```
$ cat /etc/ld.so.conf.d/x86_64-linux-gnu.conf
# Multiarch support
/lib/x86_64-linux-gnu
/usr/lib/x86_64-linux-gnu
```

El comando `ldconfig` se encarga de leer estos archivos de configuración, creando el conjunto de enlaces simbólicos antes mencionados que ayudan a localizar las bibliotecas individuales y finalmente a actualizar el archivo de caché `/etc/ld.so.cache`. Por lo tanto, `ldconfig` debe ejecutarse cada vez que se agregan o actualizan archivos de configuración.

Las opciones útiles para `ldconfig` son:

`-v`, `--verbose`

Muestra los números de versión de la biblioteca, el nombre de cada directorio y los enlaces que se crean:

```
$ sudo ldconfig -v
/usr/local/lib:
/lib/x86_64-linux-gnu:
    libnss_myhostname.so.2 -> libnss_myhostname.so.2
    libfuse.so.2 -> libfuse.so.2.9.7
    libidn.so.11 -> libidn.so.11.6.16
    libnss_mdns4.so.2 -> libnss_mdns4.so.2
    libparted.so.2 -> libparted.so.2.0.1
    (...)
```

Así podemos ver, por ejemplo, cómo se vincula `libfuse.so.2` con el archivo de objeto compartido real `libfuse.so.2.9.7`.

`-p`, `--print-cache`

Imprime las listas de directorios y bibliotecas candidatas almacenadas en la caché actual:

```
$ sudo ldconfig -p
1094 libs found in the cache `/etc/ld.so.cache'
    libzvbi.so.0 (libc6,x86-64) => /usr/lib/x86_64-linux-gnu/libzvbi.so.0
    libzvbi-chains.so.0 (libc6,x86-64) => /usr/lib/x86_64-linux-gnu/libzvbi-
chains.so.0
    libzmq.so.5 (libc6,x86-64) => /usr/lib/x86_64-linux-gnu/libzmq.so.5
    libzeitgeist-2.0.so.0 (libc6,x86-64) => /usr/lib/x86_64-linux-
gnu/libzeitgeist-2.0.so.0
    (...)
```

Observe cómo la cache usa el nombre completo del soname en el enlace:

```
$ sudo ldconfig -p |grep libfuse
      libfuse.so.2 (libc6,x86-64) => /lib/x86_64-linux-gnu/libfuse.so.2
```

Si hacemos una lista larga de `/lib/x86_64-linux-gnu/libfuse.so.2`, encontraremos la referencia al archivo de objeto compartido real `libfuse.so.2.9.7` que está almacenado en el mismo directorio:

```
$ ls -l /lib/x86_64-linux-gnu/libfuse.so.2
lrwxrwxrwx 1 root root 16 Aug 21  2018 /lib/x86_64-linux-gnu/libfuse.so.2 ->
libfuse.so.2.9.7
```

> **NOTE** Como requiere acceso de escritura a `/etc/ld.so.cache` (propiedad de root), debe ser root o usar `sudo` para invocar `ldconfig`. Para obtener más información sobre los interruptores `ldconfig`, consulte su página de manual.

Además de los archivos de configuración descritos anteriormente, la variable de entorno `LD_LIBRARY_PATH` se puede usar para agregar nuevas rutas para bibliotecas compartidas temporalmente. Está formado por un conjunto de directorios separados por dos puntos (`:`) donde se buscan las bibliotecas. Para agregar, por ejemplo, `/usr/local/mylib` a la ruta de la biblioteca en la sesión de shell actual, puede teclear:

```
$ LD_LIBRARY_PATH=/usr/local/mylib
```

Ahora puede verificar su valor:

```
$ echo $LD_LIBRARY_PATH
/usr/local/mylib
```

Para agregar `/usr/local/mylib` a la ruta de la biblioteca en la sesión de shell actual y exportarlo a todos los procesos secundarios generados desde ese shell, debe usar el siguiente comando:

```
$ export LD_LIBRARY_PATH=/usr/local/mylib
```

Para eliminar la variable de entorno `LD_LIBRARY_PATH`, simplemente utilice el siguiente comando:

```
$ unset LD_LIBRARY_PATH
```

Para que los cambios sean permanentes, usar el siguiente comando.

```
export LD_LIBRARY_PATH=/usr/local/mylib
```

en uno de los scripts de inicialización de Bash como `/etc/bash.bashrc` o `~/.bashrc`.

> **NOTE** `LD_LIBRARY_PATH` es para las bibliotecas compartidas lo que `PATH` es para los ejecutables. Para obtener más información sobre variables de entorno y configuración de shell, consulte las lecciones respectivas.

Buscando las dependencias de un ejecutable particular

Para buscar las bibliotecas compartidas requeridas por un programa específico, use el comando `ldd` seguido de la ruta absoluta al programa. El resultado muestra la ruta del archivo de la biblioteca compartida, así como la dirección de memoria hexadecimal en la que se carga:

```
$ ldd /usr/bin/git
    linux-vdso.so.1 =>  (0x00007ffcbb310000)
    libpcre.so.3 => /lib/x86_64-linux-gnu/libpcre.so.3 (0x00007f18241eb000)
    libz.so.1 => /lib/x86_64-linux-gnu/libz.so.1 (0x00007f1823fd1000)
    libresolv.so.2 => /lib/x86_64-linux-gnu/libresolv.so.2 (0x00007f1823db6000)
    libpthread.so.0 => /lib/x86_64-linux-gnu/libpthread.so.0 (0x00007f1823b99000)
    librt.so.1 => /lib/x86_64-linux-gnu/librt.so.1 (0x00007f1823991000)
    libc.so.6 => /lib/x86_64-linux-gnu/libc.so.6 (0x00007f18235c7000)
    /lib64/ld-linux-x86-64.so.2 (0x00007f182445b000)
```

Del mismo modo, usamos `ldd` para buscar las dependencias de un objeto compartido:

```
$ ldd /lib/x86_64-linux-gnu/libc.so.6
    /lib64/ld-linux-x86-64.so.2 (0x00007fbfed578000)
    linux-vdso.so.1 (0x00007fffb7bf5000)
```

Con la opción `-u` (o `--unused`) `ldd` imprime las dependencias directas no utilizadas (si existen):

```
$ ldd -u /usr/bin/git
Unused direct dependencies:
    /lib/x86_64-linux-gnu/libz.so.1
    /lib/x86_64-linux-gnu/libpthread.so.0
    /lib/x86_64-linux-gnu/librt.so.1
```

La razón de las dependencias no utilizadas está relacionada con las opciones utilizadas por el vinculador al construir el binario. Aunque el programa no necesita una biblioteca no utilizada, todavía estaba vinculado y etiquetado como `NEEDED` en la información sobre el archivo objeto. Puede investigar esto usando comandos como `readelf` u `objdump`, que pronto usará en el ejercicio de exploración.

Ejercicios Guiados

1. Divida los siguientes nombres de bibliotecas compartidas en sus partes:

Nombre completo del archivo	Nombre de la biblioteca	*so* sufijo	Número de versión
`linux-vdso.so.1`			
`libprocps.so.6`			
`libdl.so.2`			
`libc.so.6`			
`libsystemd.so.0`			
`ld-linux-x86-64.so.2`			

2. Ha desarrollado un software y desea agregar un nuevo directorio de biblioteca compartida a su sistema (`/opt/lib/mylib`). Escriba su ruta absoluta en un archivo llamado `mylib.conf`.
   - ¿En qué directorio debe almacenar este archivo?
   - ¿Qué comando debe ejecutar para que los cambios sean totalmente efectivos?

3. ¿Qué comando usaría para listar las bibliotecas compartidas requeridas por `kill`?

Ejercicios Exploratorios

1. `objdump` es una utilidad de línea de comandos que muestra información de archivos de objetos. Compruebe si está instalado en su sistema con `which objdump`. Si no es así, instálelo.
   - Use `objdump` con `-p` (o `--private-headers`) y `grep` para imprimir las dependencias de `glibc`:

   - Use `objdump` con `-p` (o `--private-headers`) y `grep` para imprimir el soname de `glibc`:

   - Use `objdump` con `-p` (o `--private-headers`) y `grep` para imprimir las dependencias de Bash:

Resumen

En esta lección aprendimos:

- ¿Qué es una biblioteca compartida (o dinámica)?
- Las diferencias entre bibliotecas compartidas y estáticas.
- Los nombres de las bibliotecas compartidas (*sonames*).
- Las ubicaciones preferidas para bibliotecas compartidas en un sistema Linux, como `/lib` o `/usr/lib`.
- El propósito del enlazador dinámico `ld.so` (o `ld-linux.so`).
- ¿Cómo configurar rutas compartidas de la biblioteca mediante archivos en `/etc/` como `ld.so.conf` o los del directorio `ld.so.conf.d`?
- ¿Cómo configurar rutas de biblioteca compartidas mediante la variable de entorno `LD_LIBRARY_PATH`?
- ¿Cómo buscar dependencias de bibliotecas ejecutables y compartidas?

Comandos utilizados en esta lección:

`ls`
: Lista el contenido de un directorio.

`cat`
: Concatena archivos e imprime en la salida estándar.

`sudo`
: Hace que el superusuario ejecute el comando con privilegios administrativos.

`ldconfig`
: Configura enlaces de tiempo de ejecución del vinculador dinámico.

`echo`
: Muestra el valor de la variable de entorno.

`export`
: Valor de exportación de la variable de entorno a shells secundarios.

`unset`
: Elimina variables de entorno.

`ldd`
: Imprime dependencias de objetos compartidos de un programa.

`readelf`
: Muestra información sobre archivos ELF (ELF significa *executable and linkable format*).

`objdump`
: Imprime información de archivos de objetos.

Respuestas a los ejercicios guiados

1. Divida los siguientes nombres de bibliotecas compartidas en sus partes:

Nombre completo del archivo	Nombre de la biblioteca	*so* sufijo	Número de versión
`linux-vdso.so.1`	`linux-vdso`	`so`	`1`
`libprocps.so.6`	`libprocps`	`so`	`6`
`libdl.so.2`	`libdl`	`so`	`2`
`libc.so.6`	`libc`	`so`	`6`
`libsystemd.so.0`	`libsystemd`	`so`	`0`
`ld-linux-x86-64.so.2`	`ld-linux-x86-64`	`so`	`2`

2. Ha desarrollado un software y desea agregar un nuevo directorio de biblioteca compartida a su sistema (`/opt/lib/mylib`). Escriba su ruta absoluta en un archivo llamado `mylib.conf`.
   - ¿En qué directorio debe almacenar este archivo?

     `/etc/ld.so.conf.d`

   - ¿Qué comando debe ejecutar para que los cambios sean totalmente efectivos?

     `ldconfig`

3. ¿Qué comando usaría para listar las bibliotecas compartidas requeridas por `kill`?

   `ldd /bin/kill`

Respuestas a ejercicios exploratorios

1. `objdump` es una utilidad de línea de comandos que muestra información de archivos de objetos. Compruebe si está instalado en su sistema con `which objdump`. Si no es así, instálelo.
   - Use `objdump` con el `-p` (o `--private-headers`) y `grep` para imprimir las dependencias de `glibc`:

     ```
     objdump -p /lib/x86_64-linux-gnu/libc.so.6 | grep NEEDED
     ```

   - Use `objdump` con el `-p` (o `--private-headers`) y `grep` para imprimir el soname de `glibc`:

     ```
     objdump -p /lib/x86_64-linux-gnu/libc.so.6 | grep SONAME
     ```

   - Use `objdump` con el `-p` (o `--private-headers`) y `grep` para imprimir las dependencias de Bash:

     ```
     objdump -p /bin/bash | grep NEEDED
     ```



102.4 Gestión de paquetes Debian

Referencia al objetivo del LPI

LPIC-1 v5, Exam 101, Objective 102.4

Importancia

3

Áreas de conocimiento clave

- Instalar, actualizar y desinstalar paquetes binarios de Debian.
- Encontrar paquetes que contengan archivos o librerías específicos (estén o no instalados).
- Obtener información del paquete como la versión, contenido, dependencias, integridad del paquete y estado de la instalación (tanto si el paquete está instalado como si no lo está).
- Conocimientos de apt.

Lista parcial de archivos, términos y utilidades

- `/etc/apt/sources.list`
- `dpkg`
- `dpkg-reconfigure`
- `apt-get`
- `apt-cache`



102.4 Lección 1

Certificación:	LPIC-1
Versión:	5.0
Tema:	102 Instalación de Linux y Administración de Paquetes
Objectivo:	102.4 Gestión de paquetes en Debian
Lección:	1 de 1

Introducción

Hace mucho tiempo, cuando Linux todavía estaba en su infancia, la forma más común de distribuir software era un archivo comprimido (generalmente un archivo `.tar.gz`) con código fuente, que usted mismo debía desempacar y compilar.

Sin embargo, a medida que crecía la cantidad y la complejidad del software, se hizo evidente la necesidad de una forma de distribuir el software precompilado. Después de todo, no todos tenían los recursos, tanto en tiempo como en potencia informática, para compilar grandes proyectos como el núcleo (Kernel) de Linux o un servidor X.

Pronto, crecieron los esfuerzos para estandarizar una forma de distribuir estos “paquetes” de software, y nacieron los primeros administradores de paquetes. Estas herramientas facilitaron mucho la instalación, configuración o eliminación de software de un sistema.

Uno de ellos fue el formato de paquete Debian (`.deb`) y su herramienta de paquetería (`dpkg`). Hoy en día, se usan ampliamente no solo en Debian, sino también en sus derivados, como Ubuntu y los derivados de él.

Otra herramienta de administración de paquetes que es popular en los sistemas basados en Debian es *Advanced Package Tool* (`apt`), que puede optimizar muchos de los aspectos de la instalación, mantenimiento y eliminación de paquetes, lo que lo hace mucho más fácil.

En esta lección, aprenderemos cómo usar tanto `dpkg` como `apt` para obtener, instalar, mantener y eliminar software en un sistema Linux basado en Debian.

La herramienta de paquetería en Debian (dpkg)

La herramienta *Debian Package* (`dpkg`) es la utilidad esencial para instalar, configurar, mantener y eliminar paquetes de software en sistemas basados en Debian. La operación más básica es instalar un paquete `.deb`, que se puede hacer con:

```
# dkpg -i PACKAGENAME
```

Donde `PACKAGENAME` es el nombre del archivo `.deb` que desea instalar.

Las actualizaciones de paquetes se manejan de la misma manera. Antes de instalar un paquete, `dpkg` verificará si ya existe una versión anterior en el sistema. Si es así, el paquete se actualizará a la nueva versión. Si no, se instalará una copia nueva.

Manejo de dependencias

La mayoría de las veces, un paquete puede depender de otros para que funcionen. Por ejemplo, un editor de imágenes puede necesitar bibliotecas para abrir archivos JPEG, u otra utilidad puede necesitar un kit de herramientas como Qt o GTK para su interfaz de usuario.

`dpkg` verificará si esas dependencias están instaladas en su sistema y no podrá instalar el paquete si no lo están. En este caso, `dpkg` listará qué paquetes faltan. Sin embargo, no puede resolver dependencias por sí mismo. Depende del usuario encontrar los paquetes `.deb` con las dependencias correspondientes e instalarlos.

En el siguiente ejemplo, el usuario intenta instalar el paquete del editor de video OpenShot, pero faltan algunas dependencias:

```
# dpkg -i openshot-qt_2.4.3+dfsg1-1_all.deb
(Reading database ... 269630 files and directories currently installed.)
Preparing to unpack openshot-qt_2.4.3+dfsg1-1_all.deb ...
Unpacking openshot-qt (2.4.3+dfsg1-1) over (2.4.3+dfsg1-1) ...
dpkg: dependency problems prevent configuration of openshot-qt:
 openshot-qt depends on fonts-cantarell; however:
  Package fonts-cantarell is not installed.
 openshot-qt depends on python3-openshot; however:
  Package python3-openshot is not installed.
 openshot-qt depends on python3-pyqt5; however:
  Package python3-pyqt5 is not installed.
 openshot-qt depends on python3-pyqt5.qtsvg; however:
  Package python3-pyqt5.qtsvg is not installed.
 openshot-qt depends on python3-pyqt5.qtwebkit; however:
  Package python3-pyqt5.qtwebkit is not installed.
 openshot-qt depends on python3-zmq; however:
  Package python3-zmq is not installed.

dpkg: error processing package openshot-qt (--install):
 dependency problems - leaving unconfigured
Processing triggers for mime-support (3.60ubuntu1) ...
Processing triggers for gnome-menus (3.32.0-1ubuntu1) ...
Processing triggers for desktop-file-utils (0.23-4ubuntu1) ...
Processing triggers for hicolor-icon-theme (0.17-2) ...
Processing triggers for man-db (2.8.5-2) ...
Errors were encountered while processing:
 openshot-qt
```

Como se muestra, OpenShot depende de los paquetes `fonts-cantarell`, `python3-openshot`, `python3-pyqt5`, `python3-pyqt5.qtsvg`, `python3-pyqt5.qtwebkit` y `python3-zmq`. Todos ellos deben instalarse previamente para que la instalación de OpenShot pueda tener éxito.

Eliminar Paquetes

Para eliminar un paquete, pase el parámetro `-r` a `dpkg`, seguido del nombre del paquete. Por ejemplo, el siguiente comando eliminará el paquete `unrar` del sistema:

```
# dpkg -r unrar
(Reading database ... 269630 files and directories currently installed.)
Removing unrar (1:5.6.6-2) ...
Processing triggers for man-db (2.8.5-2) ...
```

La operación de eliminación también ejecuta una verificación de dependencias, y un paquete no se puede eliminar a menos que también se elimine cualquier otro paquete que dependa de él. Si intenta hacerlo, recibirá un mensaje de error como el siguiente:

```
# dpkg -r p7zip
dpkg: dependency problems prevent removal of p7zip:
 winetricks depends on p7zip; however:
  Package p7zip is to be removed.
 p7zip-full depends on p7zip (= 16.02+dfsg-6).

dpkg: error processing package p7zip (--remove):
 dependency problems - not removing
Errors were encountered while processing:
 p7zip
```

Puede pasar varios nombres de paquetes a `dpkg -r`, por lo que se eliminarán todos a la vez.

Cuando se elimina un paquete, los archivos de configuración correspondientes se dejan en el sistema. Si desea eliminar todo lo relacionado con el paquete, use la opción `-P` (purgar) en lugar de `-r`.

NOTE

Puede forzar la instalación o eliminación de un paquete a través de *dpkg*, incluso si no se cumplen las dependencias, agregando el parámetro `--force` como por ejemplo `dpkg -i --force PACKAGENAME`. Sin embargo, hacerlo probablemente dejará el paquete instalado, o incluso su sistema, en un estado incorrecto con paquetes rotos. *No* use `--force` a menos que esté absolutamente seguro de lo que está haciendo.

Obtener Información de Paquetes

Para obtener información sobre un paquete `.deb`, como su versión, arquitectura, mantenedor, dependencias y más, use el comando `dpkg` con el parámetro `-I`, seguido del nombre de archivo del paquete que desea inspeccionar:

```
# dpkg -I google-chrome-stable_current_amd64.deb
 new Debian package, version 2.0.
 size 59477810 bytes: control archive=10394 bytes.
    1222 bytes,    13 lines      control
   16906 bytes,   457 lines   *  postinst             #!/bin/sh
   12983 bytes,   344 lines   *  postrm               #!/bin/sh
    1385 bytes,    42 lines   *  prerm                #!/bin/sh
 Package: google-chrome-stable
 Version: 76.0.3809.100-1
 Architecture: amd64
 Maintainer: Chrome Linux Team <chromium-dev@chromium.org>
 Installed-Size: 205436
 Pre-Depends: dpkg (>= 1.14.0)
 Depends: ca-certificates, fonts-liberation, libappindicator3-1, libasound2 (>=
1.0.16), libatk-bridge2.0-0 (>= 2.5.3), libatk1.0-0 (>= 2.2.0), libatspi2.0-0 (>=
2.9.90), libc6 (>= 2.16), libcairo2 (>= 1.6.0), libcups2 (>= 1.4.0), libdbus-1-3
(>= 1.5.12), libexpat1 (>= 2.0.1), libgcc1 (>= 1:3.0), libgdk-pixbuf2.0-0 (>=
2.22.0), libglib2.0-0 (>= 2.31.8), libgtk-3-0 (>= 3.9.10), libnspr4 (>= 2:4.9-2~),
libnss3 (>= 2:3.22), libpango-1.0-0 (>= 1.14.0), libpangocairo-1.0-0 (>= 1.14.0),
libuuid1 (>= 2.16), libx11-6 (>= 2:1.4.99.1), libx11-xcb1, libxcb1 (>= 1.6),
libxcomposite1 (>= 1:0.3-1), libxcursor1 (>> 1.1.2), libxdamage1 (>= 1:1.1),
libxext6, libxfixes3, libxi6 (>= 2:1.2.99.4), libxrandr2 (>= 2:1.2.99.3),
libxrender1, libxss1, libxtst6, lsb-release, wget, xdg-utils (>= 1.0.2)
 Recommends: libu2f-udev
 Provides: www-browser
 Section: web
 Priority: optional
 Description: The web browser from Google
  Google Chrome is a browser that combines a minimal design with sophisticated
technology to make the web faster, safer, and easier.
```

Listar paquetes instalados y contenido del paquete

Para obtener una lista de cada paquete instalado en su sistema, use la opción `--get-selections`, como por ejemplo `dpkg --get-selections`. También puede obtener una lista de cada archivo instalado por un paquete específico pasando el parámetro `-L PACKAGENAME` a `dpkg`, como se muestra a continuación:


```
# dpkg -L unrar
/.
/usr
/usr/bin
/usr/bin/unrar-nonfree
/usr/share
/usr/share/doc
/usr/share/doc/unrar
/usr/share/doc/unrar/changelog.Debian.gz
/usr/share/doc/unrar/copyright
/usr/share/man
/usr/share/man/man1
/usr/share/man/man1/unrar-nonfree.1.gz
```

Averiguar qué paquete posee un archivo específico

A veces es posible que necesite averiguar qué paquete posee un archivo específico en su sistema. Puede hacerlo utilizando la utilidad `dpkg-query`, seguida del parámetro `-S` y la ruta al archivo en cuestión:

```
# dpkg-query -S /usr/bin/unrar-nonfree
unrar: /usr/bin/unrar-nonfree
```

Reconfigurar Paquetes Instalados

Cuando se instala un paquete, hay un paso de configuración llamado *post-install* donde se ejecuta un script para configurar todo lo necesario para que el software se ejecute, como permisos, ubicación de archivos de configuración, etc. Esto también puede generar algunas preguntas de configuración al usuario para establecer preferencias sobre cómo se ejecutará el software.

A veces, debido a un archivo de configuración dañado o con formato incorrecto, es posible que desee restaurar las configuraciones de un paquete a su estado "funcional". O puede que desee cambiar las respuestas que dio a las preguntas de configuración inicial. Para hacer esto, ejecute la utilidad `dpkg-reconfigure`, seguida del nombre del paquete.

Este programa realizará una copia de seguridad de los archivos de configuración antiguos, descomprimirá los nuevos en los directorios correctos y ejecutará el script *post-install* proporcionado por el paquete, como si el paquete se hubiera instalado por primera vez. Intente reconfigurar el paquete `tzdata` con el siguiente ejemplo:

```
# dpkg-reconfigure tzdata
```

Herramienta de Paquetería Avanzada (apt)

Advanced Package Tool (APT) es un sistema de administración de paquetes, que incluye un conjunto de herramientas, que simplifica enormemente la instalación, actualización, eliminación y administración de paquetes. APT proporciona características como capacidades de búsqueda avanzada y resolución de dependencias automática.

APT no es un “sustituto” de `dpkg`. Puede considerarlo como una “Interfaz (front end)”, que optimiza las operaciones y llena los vacíos de la funcionalidad `dpkg`, como la resolución de dependencias.

APT trabaja en conjunto con los repositorios de software que contienen los paquetes disponibles para instalar. Dichos repositorios pueden ser un servidor local o remoto o (menos común) incluso un disco CD-ROM.

Las distribuciones de Linux, como Debian y Ubuntu, mantienen sus propios repositorios, y los desarrolladores o grupos de usuarios pueden mantener otros repositorios para proporcionar software que no está disponible en los principales repositorios de distribución.

Existen muchas utilidades que interactúan con APT, siendo las principales:

`apt-get`
: Se utiliza para descargar, instalar, actualizar o eliminar paquetes del sistema.

`apt-cache`
: Se utiliza para realizar operaciones, como búsquedas, en el índice de paquetes.

`apt-file`
: se utiliza para buscar archivos dentro de los paquetes.

También hay una utilidad “más amigable” llamada simplemente `apt`, que combina las opciones más utilizadas de `apt-get` y `apt-cache` en una utilidad. Muchos de los comandos para `apt` son los mismos que para `apt-get`, por lo que en muchos casos son intercambiables. Sin embargo, dado que `apt` puede no estar instalado en un sistema, se recomienda aprender a usar `apt-get` y `apt-cache`.

NOTE `apt` y `apt-get` pueden requerir una conexión de red, porque los paquetes y los índices de paquetes pueden necesitar descargarse de un servidor remoto.

Actualización de los Índices de Paquetes

Antes de instalar o actualizar un software con APT, se recomienda actualizar primero el índice de paquetes para recuperar información sobre paquetes nuevos y actualizados. Esto se hace con el comando `apt-get`, seguido del parámetro `update`:

```
# apt-get update
Ign:1 http://dl.google.com/linux/chrome/deb stable InRelease
Hit:2 https://repo.skype.com/deb stable InRelease
Hit:3 http://us.archive.ubuntu.com/ubuntu disco InRelease
Hit:4 http://repository.spotify.com stable InRelease
Hit:5 http://dl.google.com/linux/chrome/deb stable Release
Hit:6 http://apt.pop-os.org/proprietary disco InRelease
Hit:7 http://ppa.launchpad.net/system76/pop/ubuntu disco InRelease
Hit:8 http://us.archive.ubuntu.com/ubuntu disco-security InRelease
Hit:9 http://us.archive.ubuntu.com/ubuntu disco-updates InRelease
Hit:10 http://us.archive.ubuntu.com/ubuntu disco-backports InRelease
Reading package lists... Done
```

TIP En lugar de `apt-get update`, también puede utilizar `apt update`.

Instalar y Remover Paquetes

Con los índices de paquetes actualizados, ahora puede instalar un paquete. Esto se hace con `apt-get install`, seguido del nombre del paquete que desea instalar:

```
# apt-get install xournal
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following NEW packages will be installed:
  xournal
0 upgraded, 1 newly installed, 0 to remove and 75 not upgraded.
Need to get 285 kB of archives.
After this operation, 1041 kB of additional disk space will be used.
```

Del mismo modo, para eliminar un paquete, use `apt-get remove`, seguido del nombre del paquete:

```
# apt-get remove xournal
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following packages will be REMOVED:
  xournal
0 upgraded, 0 newly installed, 1 to remove and 75 not upgraded.
After this operation, 1041 kB disk space will be freed.
Do you want to continue? [Y/n]
```

Tenga en cuenta que al instalar o eliminar paquetes, APT hará una resolución de dependencias automática. Esto significa que cualquier paquete adicional que necesite el paquete que está instalando *también se instalará*, y que los paquetes que dependen del paquete que está eliminando *también se eliminarán*. APT siempre mostrará lo que se instalará o eliminará y le preguntará si desea continuar:

```
# apt-get remove p7zip
Reading package lists... Done
Building dependency tree
The following packages will be REMOVED:
  android-libbacktrace android-libunwind android-libutils
  android-libziparchive android-sdk-platform-tools fastboot p7zip p7zip-full
0 upgraded, 0 newly installed, 8 to remove and 75 not upgraded.
After this operation, 6545 kB disk space will be freed.
Do you want to continue? [Y/n]
```

Tenga en cuenta que cuando se elimina un paquete, los archivos de configuración correspondientes quedan en el sistema. Para eliminar el paquete *y* cualquier archivo de configuración, use el parámetro `purge` en lugar de `remove` o el parámetro `remove` con la opción `--purge`:

```
# apt-get purge p7zip
```

o

```
# apt-get remove --purge p7zip
```

TIP También puede utilizar `apt install` y `apt remove`.

Reparar Dependencias Rotas

Es posible tener “dependencias rotas” en un sistema. Esto significa que uno o más de los paquetes instalados dependen de otros paquetes que no se han instalado o que ya no están presentes. Esto puede suceder debido a un error de APT o debido a un paquete instalado manualmente.

Para resolver esto, use el comando `apt-get install -f`. Esto intentará “arreglar” los paquetes rotos instalando las dependencias que faltan, asegurando que todos los paquetes sean consistentes nuevamente.

TIP También puede usar `apt install -f`.

Actualizar Paquetes

APT se puede utilizar para actualizar automáticamente cualquier paquete instalado a las últimas versiones disponibles desde los repositorios. Esto se hace con el comando `apt-get upgrade`. Antes de ejecutarlo, primero actualice el índice de paquetes con `apt-get update`:

```
# apt-get update
Hit:1 http://us.archive.ubuntu.com/ubuntu disco InRelease
Hit:2 http://us.archive.ubuntu.com/ubuntu disco-security InRelease
Hit:3 http://us.archive.ubuntu.com/ubuntu disco-updates InRelease
Hit:4 http://us.archive.ubuntu.com/ubuntu disco-backports InRelease
Reading package lists... Done

# apt-get upgrade
Reading package lists... Done
Building dependency tree
Reading state information... Done
Calculating upgrade... Done
The following packages have been kept back:
  gnome-control-center
The following packages will be upgraded:
  cups cups-bsd cups-client cups-common cups-core-drivers cups-daemon
  cups-ipp-utils cups-ppdc cups-server-common firefox-locale-ar (...)

74 upgraded, 0 newly installed, 0 to remove and 1 not upgraded.
Need to get 243 MB of archives.
After this operation, 30.7 kB of additional disk space will be used.
Do you want to continue? [Y/n]
```

El resumen en la parte inferior de la salida muestra cuántos paquetes se actualizarán, cuántos se

instalarán, eliminarán o retendrán, el tamaño total de descarga y cuánto espacio de disco adicional será necesario para completar la operación. Para completar la actualización, simplemente responda `Y` y espere a que `apt-get` termine la tarea.

Para actualizar un solo paquete, simplemente ejecute `apt-get upgrade` seguido del nombre del paquete. Como en `dpkg`, `apt-get` primero verificará si está instalada una versión anterior del paquete. Si es así, el paquete se actualizará a la versión más nueva disponible en el repositorio. Si no, se instalará una copia nueva.

TIP También puede utilizar `apt upgrade` y `apt update`.

La Caché Local

Cuando instala o actualiza un paquete, el archivo `.deb` correspondiente se descarga en un directorio de caché local antes de instalar el paquete. Por defecto, este directorio es `/var/cache/apt/archives`. Los archivos descargados parcialmente se copian a `/var/cache/apt/archives/partial/`.

A medida que instala y actualiza paquetes, el directorio de caché puede ser bastante grande. Para recuperar espacio, puede vaciar la caché utilizando el comando `apt-get clean`. Esto eliminará el contenido de los directorios `/var/cache/apt/archives` y `/var/cache/apt/archives/partial/`.

TIP También puede utilizar `apt clean`.

Buscar Paquetes

La utilidad `apt-cache` se puede usar para realizar operaciones en el índice de paquetes, como buscar un paquete específico o listar qué paquetes contienen un archivo específico.

Para realizar una búsqueda, use `apt-cache search` seguido de un patrón de búsqueda. El resultado será una lista de cada paquete que contiene el patrón, ya sea en el nombre del paquete, la descripción o los archivos proporcionados.

```
# apt-cache search p7zip
liblzma-dev - XZ-format compression library - development files
liblzma5 - XZ-format compression library
forensics-extra - Forensics Environment - extra console components (metapackage)
p7zip - 7zr file archiver with high compression ratio
p7zip-full - 7z and 7za file archivers with high compression ratio
p7zip-rar - non-free rar module for p7zip
```

En el ejemplo anterior, la entrada `liblzma5 - XZ-format compression library` no parece coincidir con el patrón. Sin embargo, si mostramos la información completa, incluida la descripción, del paquete usando el parámetro `show`, encontraremos el patrón allí:

```
# apt-cache show liblzma5
Package: liblzma5
Architecture: amd64
Version: 5.2.4-1
Multi-Arch: same
Priority: required
Section: libs
Source: xz-utils
Origin: Ubuntu
Maintainer: Ubuntu Developers <ubuntu-devel-discuss@lists.ubuntu.com>
Original-Maintainer: Jonathan Nieder <jrnieder@gmail.com>
Bugs: https://bugs.launchpad.net/ubuntu/+filebug
Installed-Size: 259
Depends: libc6 (>= 2.17)
Breaks: liblzma2 (<< 5.1.1alpha+20110809-3~)
Filename: pool/main/x/xz-utils/liblzma5_5.2.4-1_amd64.deb
Size: 92352
MD5sum: 223533a347dc76a8cc9445cfc6146ec3
SHA1: 8ed14092fb1caecfebc556fda0745e1e74ba5a67
SHA256: 01020b5a0515dbc9a7c00b464a65450f788b0258c3fbb733ecad0438f5124800
Homepage: https://tukaani.org/xz/
Description-en: XZ-format compression library
 XZ is the successor to the Lempel-Ziv/Markov-chain Algorithm
 compression format, which provides memory-hungry but powerful
 compression (often better than bzip2) and fast, easy decompression.
 .
 The native format of liblzma is XZ; it also supports raw (headerless)
 streams and the older LZMA format used by lzma. (For 7-Zip's related
 format, use the p7zip package instead.)
```

Puede usar *expresiones regulares* con el patrón de búsqueda, lo que permite búsquedas muy complejas (y precisas). Sin embargo, este tema está fuera del alcance de esta lección.

TIP También puede utilizar `apt search` en lugar de `apt-cache search` y `apt show` en lugar de `apt-cache show`.

La Lista de Fuentes

APT utiliza una lista de fuentes para saber de dónde obtener paquetes. Esta lista se almacena en el archivo `sources.list`, ubicado dentro del directorio `/etc/apt`. Este archivo se puede editar directamente con un editor de texto, como `vi`, `pico` o `nano`, o con utilidades gráficas como `aptitude` o `synaptic`.

Una línea típica dentro de `sources.list` se ve así:

```
deb http://us.archive.ubuntu.com/ubuntu/ disco main restricted universe multiverse
```

La sintaxis es tipo de archivo, URL, distribución y uno o más componentes, donde:

Tipo de archivo

Un repositorio puede contener paquetes con software listo para ejecutar (paquetes binarios, descrito como `deb`) o con el código fuente de este software (paquetes fuente, descrito como `deb-src`). El ejemplo anterior proporciona paquetes binarios.

URL

La URL del repostorio.

Distribución

El nombre (o nombre en clave) de la distribución para la que se proporcionan los paquetes. Un repositorio puede alojar paquetes para múltiples distribuciones. En el ejemplo anterior, `disco` es el nombre en clave de Ubuntu 19.04 *Disco Dingo*.

Componentes

Cada componente representa un conjunto de paquetes. Estos componentes pueden ser diferentes en diferentes distribuciones de Linux. Por ejemplo, en Ubuntu y derivados, son:

`main`

contiene paquetes de código abierto con soporte oficial.

`restricted`

contiene software de código cerrado con soporte oficial, como controladores de dispositivo para tarjetas gráficas, por ejemplo.

`universe`

contiene software de código abierto mantenido por la comunidad.

`multiverse`
: contiene software no compatible, de código cerrado o con patente gravada.

En Debian, los componentes principales son:

`main`
: consiste en paquetes que cumplen con las *Directrices de software libre de Debian* (DFSG), que no dependen de software fuera de esta área para operar. Los paquetes incluidos aquí se consideran parte de la distribución Debian.

`contrib`
: contiene paquetes compatibles con DFSG, pero que dependen de otros paquetes que no están en `main`.

`non-free`
: contiene paquetes que no son compatibles con DFSG.

`security`
: contiene actualizaciones de seguridad.

`backports`
: contiene versiones más recientes de paquetes que están en `main`. El ciclo de desarrollo de las versiones estables de Debian es bastante largo (alrededor de dos años), y esto asegura que los usuarios puedan obtener los paquetes más actualizados sin tener que modificar el repositorio principal `main`.

NOTE Puede aprender más sobre las *Debian Free Software Guidelines* en: https://www.debian.org/social_contract#guidelines

Para agregar un nuevo repositorio de paquetes, simplemente puede agregar la línea correspondiente (generalmente proporcionada por el responsable del repositorio) al final de `sources.list`, guarde el archivo y vuelva a cargar el índice del paquete con `apt-get update` . Después de eso, los paquetes en el nuevo repositorio estarán disponibles para la instalación usando `apt-get install`.

Tenga en cuenta que las líneas que comienzan con el carácter `#` se consideran comentarios y se ignoran.

El Directorio `/etc/apt/sources.list.d`

Dentro del directorio `/etc/apt/sources.list.d` puede agregar archivos con repositorios adicionales para ser utilizados por APT, sin la necesidad de modificar el archivo principal `/etc/apt/sources.list`. Estos son archivos de texto simples, con la misma sintaxis descrita anteriormente y la extensión de archivo `.list`.

A continuación puede ver el contenido de un archivo llamado `/etc/apt/sources.list.d/buster-backports.list`:

```
deb http://deb.debian.org/debian buster-backports main contrib non-free
deb-src http://deb.debian.org/debian buster-backports main contrib non-free
```

Listar el contenido de paquetes y búsqueda de archivos

Una utilidad llamada `apt-file` puede usarse para realizar más operaciones en el índice de paquetes, como listar el contenido de un paquete o encontrar un paquete que contenga un archivo específico. Es posible que esta utilidad no esté instalada de manera predeterminada en su sistema. En este caso, generalmente puede instalarlo usando `apt-get`:

```
# apt-get install apt-file
```

Después de la instalación, deberá actualizar la caché del paquete utilizada para `apt-file`:

```
# apt-file update
```

Esto generalmente toma solo unos segundos. Después de eso, `apt-file` estará listo para usarse.

Para enumerar el contenido de un paquete, use el parámetro `list` seguido del nombre del paquete:

```
# apt-file list unrar
unrar: /usr/bin/unrar-nonfree
unrar: /usr/share/doc/unrar/changelog.Debian.gz
unrar: /usr/share/doc/unrar/copyright
unrar: /usr/share/man/man1/unrar-nonfree.1.gz
```

TIP También puede usar `apt list` en lugar de `apt-file list`.

Puede buscar un archivo en todos los paquetes utilizando el parámetro `search`, seguido del nombre del archivo. Por ejemplo, si desea saber qué paquete proporciona un archivo llamado `libSDL2.so`, puede usar:

```
# apt-file search libSDL2.so
libsdl2-dev: /usr/lib/x86_64-linux-gnu/libSDL2.so
```

La respuesta es el paquete `libsdl2-dev`, que proporciona el archivo `/usr/lib/x86_64-linux-gnu/libSDL2.so`.

La diferencia entre `apt-file search` y `dpkg-query` es que `apt-file search` también buscará paquetes desinstalados, mientras que `dpkg-query` solo puede mostrar archivos que pertenecen a un paquete instalado.

Ejercicios Guiados

1. ¿Cuál es el comando para instalar un paquete llamado `package.deb` usando `dpkg`?

2. Usando `dpkg-query`, encuentre qué paquete contiene un archivo llamado `7zr.1.gz`.

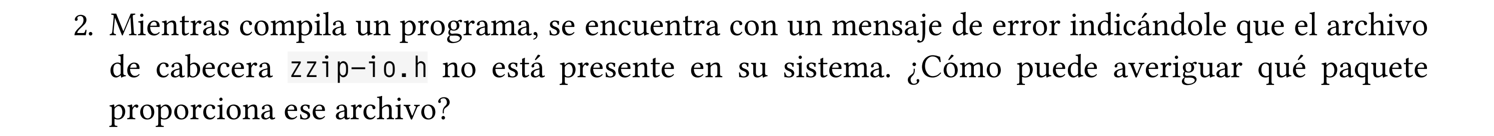
3. ¿Puede eliminar un paquete llamado `unzip` del sistema usando `dpkg -r unzip` si el paquete `file-roller` depende de él? Si no, ¿cuál sería la forma correcta de hacerlo?

4. Usando `apt-file`, ¿cómo puede averiguar qué paquete contiene el archivo `unrar`?

5. Usando `apt-cache`, ¿cuál es el comando para mostrar información para el paquete `gimp`?

Ejercicios Exploratorios

1. Considere un repositorio con paquetes fuente de Debian para la distribución `xenial`, alojado en `http://us.archive.ubuntu.com/ubuntu/` y con paquetes para el componente `universe`. ¿Cuál sería la línea correspondiente que se agregará a `/etc/apt/sources.list`?



2. Mientras compila un programa, se encuentra con un mensaje de error indicándole que el archivo de cabecera `zzip-io.h` no está presente en su sistema. ¿Cómo puede averiguar qué paquete proporciona ese archivo?

3. ¿Cómo puede ignorar una advertencia de dependencia y eliminar un paquete usando `dpkg`, incluso si hay paquetes que dependen de él en el sistema?

4. ¿Cómo puede obtener más información sobre un paquete llamado `midori` usando `apt`?

5. Antes de instalar o actualizar paquetes con `apt`, ¿qué comando se debe usar para garantizar que el índice de paquetes esté actualizado?

Resumen

En esta lección aprendimos:

- ¿Cómo usar `dpkg` para instalar y eliminar paquetes?
- ¿Cómo listar los paquetes instalados y el contenido del paquete?
- ¿Cómo reconfigurar un paquete instalado?
- ¿Qué es `apt`?, y cómo instalar, actualizar y eliminar paquetes que lo usan.
- ¿Cómo usar `apt-cache` para buscar paquetes?
- ¿Cómo funciona el archivo `/etc/apt/sources.list`?
- ¿Cómo usar `apt-file` para mostrar el contenido de un paquete?, o ¿cómo encontrar qué paquete contiene un archivo específico?

Los siguientes comandos se discutieron en esta lección:

`dpkg -i`
: Instala un paquete individual o una lista de paquetes separados por espacios.

`dpkg -r`
: Elimina un paquete o una lista de paquetes separados por espacios.

`dpkg -I`
: Inspecciona un paquete, proporcionando detalles sobre el software que instala y las dependencias necesarias.

`dpkg --get-selections`
: Enumera todos los paquetes que `dpkg` ha instalado en el sistema.

`dpkg -L`
: Imprime una lista de cada archivo que instala un paquete en particular.

`dpkg-query`
: Con un nombre de archivo especificado, este comando imprimirá el paquete que instaló el archivo.

`dpkg-reconfigure`
: Este comando volverá a ejecutar una secuencia de comandos *post-install* de paquetes para que un administrador pueda hacer ajustes de configuración a la instalación del paquete.

`apt-get update`
: Este comando actualizará el índice del paquete local para que coincida con lo que está disponible dentro de los repositorios configurados en el directorio `/etc/apt/`.

`apt-get install`
: Este comando descargará un paquete desde un repositorio remoto y lo instalará junto con sus dependencias, también se puede usar para instalar un paquete Debian que ya se ha descargado.

`apt-get remove`
: Este comando desinstalará los paquetes especificados del sistema.

`apt-cache show`
: Al igual que el comando `dpkg -I`, este comando puede usarse para mostrar detalles en un paquete específico.

`apt-cache search`
: Este comando buscará en su base de datos APT local en caché un paquete en particular.

`apt-file update`
: Este comando actualizará la caché del paquete para que el comando `apt-file` pueda consultar su contenido.

`apt-file search`
: Este comando puede buscar el nombre de un paquete que ha instalado un archivo en particular, al igual que el comando `dpkg-query`.

`apt-file list`
: Este comando se usa para listar el contenido de un paquete, al igual que el comando `dpkg -L`.

Respuestas a los ejercicios guiados

1. ¿Cuál es el comando para instalar un paquete llamado `package.deb` usando `dpkg`?

   Utilice el parámetro `-i` a `dpkg`:

   ```
   # dpkg -i package.deb
   ```

2. Usando `dpkg-query`, encuentre ¿qué paquete contiene un archivo llamado `7zr.1.gz`?

   Agregue el parámetro `-S` a `dpkg-query`:

   ```
   # dpkg-query -S 7zr.1.gz
   ```

3. ¿Puede eliminar un paquete llamado `unzip` del sistema usando `dpkg -r unzip` si el paquete `file-roller` depende de él? Si no, ¿cuál sería la forma correcta de hacerlo?

   No. `dpkg` no resolverá las dependencias y no le permitirá eliminar un paquete si otro paquete instalado depende de él. En este ejemplo, primero puede eliminar `file-roller` (suponiendo que nada depende de él) y luego eliminar `unzip`, o eliminar ambos al mismo tiempo con:

   ```
   # dpkg -r unzip file-roller
   ```

4. ¿Cómo puede averiguar qué paquete contiene el archivo `/usr/bin/unrar` utilizando la utilidad `apt-file`?

   Use el parámetro `search` seguido de la ruta (o nombre de archivo):

   ```
   # apt-file search /usr/bin/unrar
   ```

5. Usando `apt-cache`, ¿cuál es el comando para mostrar información para el paquete `gimp`?

   Use el parámetro `show` seguido del nombre del paquete:

   ```
   # apt-cache show gimp
   ```

Respuestas a ejercicios exploratorios

1. Considere un repositorio con paquetes fuente de Debian para la distribución `xenial`, alojado en `http://us.archive.ubuntu.com/ubuntu/` y con paquetes para el componente `universe`. ¿Cuál sería la línea correspondiente que se agregará a `/etc/apt/sources.list`?

   Los paquetes fuente son del tipo `deb-src`, por lo que la línea debe ser:

   ```
   deb-src http://us.archive.ubuntu.com/ubuntu/ xenial universe
   ```

   Esta línea también podría agregarse dentro de un archivo `.list` en `/etc/apt/sources.list.d/`. El nombre depende de usted, pero debe ser descriptivo, algo así como `xenial_sources.list`.

2. Mientras compila un programa, se encuentra con un mensaje de error indicándole que el archivo de cabecera `zzip-io.h` no está presente en su sistema. ¿Cómo puede averiguar qué paquete proporciona ese archivo?

   Use `apt-file search` para encontrar qué paquete contiene un archivo que no está presente en el sistema:

   ```
   # apt-file search zzip-io.h
   ```

3. ¿Cómo puede ignorar una advertencia de dependencia y eliminar un paquete usando `dpkg`, incluso si hay paquetes que dependen de él en el sistema?

   Se puede usar el parámetro `--force`, pero esto nunca se debe hacer a menos que sepa exactamente lo que está haciendo, ya que existe un gran riesgo de que su sistema quede en un estado inconsistente o “roto”.

4. ¿Cómo puede obtener más información sobre un paquete llamado `midori` usando `apt-cache`?

   Use `apt-cache show` seguido del nombre del paquete:

   ```
   # apt-cache show midori
   ```

5. Antes de instalar o actualizar paquetes con `apt-get`, ¿qué comando se debe usar para garantizar que el índice de paquetes esté actualizado?

   Se debe usar `apt-get update`. Esto descargará los últimos índices de paquetes de los repositorios

descritos en el archivo `/etc/apt/sources.list` o en el directorio `/etc/apt/sources.list.d/`.



102.5 Gestión de paquetes RPM y YUM

Referencia al objetivo del LPI

LPIC-1 v5, Exam 101, Objective 102.5

Importancia

3

Áreas de conocimiento clave

- Instalar, reinstalar, actualizar y desinstalar paquetes usando RPM, YUM y Zypper.
- Obtener información de paquetes RPM como la versión, estado, dependencias, integridad y firmas.
- Determinar qué archivos proporciona un paquete así como encontrar de qué paquete proviene un determinado archivo.
- Conocimientos de dnf.

Lista parcial de archivos, términos y utilidades

- `rpm`
- `rpm2cpio`
- `/etc/yum.conf`
- `/etc/yum.repos.d/`
- `yum`
- `zypper`



102.5 Lección 1

Certificación:	LPIC-1
Versión:	5.0
Tema:	102 Instalación de Linux y Administración de Paquetes
Objetivo:	102.5 Uso y gestión de paquetes con RPM y YUM
Lección:	1 de 1

Introducción

Hace mucho tiempo, cuando Linux todavía estaba en su infancia, la forma más común de distribuir software era a través de archivos comprimidos (generalmente como archivos `.tar.gz`) con código fuente, que desempaquetaría y compilaría usted mismo.

Sin embargo, a medida que crecía la cantidad y la complejidad del software, se hizo evidente la necesidad de una forma de distribuir el software precompilado. Después de todo, no todos tenían los recursos, tanto en tiempo como en potencia informática, para compilar grandes proyectos como el núcleo (Kernel) de Linux o un servidor X.

Pronto, crecieron los esfuerzos para estandarizar una forma de distribuir estos “paquetes” de software, y nacieron los primeros administradores de paquetes. Estas herramientas facilitaron mucho la instalación, configuración o eliminación de software de un sistema.

Uno de ellos fue el *RPM Package Manager* y su herramienta correspondiente (`rpm`), desarrollada por Red Hat. Hoy en día, se usan ampliamente no solo en Red Hat Enterprise Linux (RHEL), sino también en sus descendientes, como Fedora, CentOS y Oracle Linux, otras distribuciones como openSUSE e incluso otros sistemas operativos, como AIX de IBM.

Otras herramientas de administración de paquetes populares en las distribuciones compatibles con Red Hat son `yum` (YellowDog Updater Modified), `dnf` (Dandified YUM) y `zypper`, que pueden simplificar muchos de los aspectos de la instalación, mantenimiento y eliminación de paquetes, haciendo que gestión de paquetes mucho más fácil.

En esta lección, aprenderemos cómo usar `rpm`, `yum`, `dnf` y `zypper` para obtener, instalar, administrar y eliminar software en un sistema Linux.

NOTE A pesar de usar el mismo formato de paquete, existen diferencias internas entre las distribuciones, por lo que un paquete hecho específicamente para openSUSE podría no funcionar en un sistema RHEL, y viceversa. Cuando busque paquetes, siempre verifique la compatibilidad e intente encontrar uno adaptado a su distribución específica.

El gestor de paquetes RPM (rpm)

El gestor de paquetes RPM (`rpm`) es la herramienta esencial para administrar paquetes de software en sistemas basados en Red Hat (o derivados).

Instalar, Actualizar y Eliminar Paquetes

La operación más básica es instalar un paquete, que se puede hacer con:

```
# rpm -i PACKAGENAME
```

Donde `PACKAGENAME` es el nombre del paquete `.rpm` que desea instalar. Si hay una versión anterior de un paquete en el sistema, puede actualizar a una versión más nueva utilizando el parámetro `-U`:

```
# rpm -U PACKAGENAME
```

Si no hay instalada una versión anterior de `PACKAGENAME`, se instalará una copia nueva. Para evitar esto y *solo* actualizar un paquete *instalado*, use la opción `-F`.

En ambas operaciones, puede agregar el parámetro `-v` para obtener una salida detallada (se muestra más información durante la instalación) y `-h` para obtener signos hash (`#`) impresos como una ayuda visual para rastrear el progreso de la instalación. Se pueden combinar varios parámetros en uno, por lo que `rpm -i -v -h` es lo mismo que `rpm -ivh`.

Para eliminar un paquete instalado, pase el parámetro `-e` (como en “erase”) a `rpm`, seguido del

nombre del paquete que desea eliminar:

```
# rpm -e wget
```

Si un paquete instalado depende del paquete que se está eliminando, recibirá un mensaje de error:

```
# rpm -e unzip
error: Failed dependencies:
    /usr/bin/unzip is needed by (installed) file-roller-3.28.1-2.el7.x86_64
```

Para completar la operación, primero deberá eliminar los paquetes que dependen del que desea eliminar (en el ejemplo anterior, `file-roller`). Puede pasar varios nombres a `rpm -e` para eliminar varios paquetes a la vez.

Manejo de dependencias

La mayoría de las veces, un paquete puede depender de otros para que funcione según lo previsto. Por ejemplo, un editor de imágenes puede necesitar bibliotecas para abrir archivos JPG, o una utilidad puede necesitar un kit de herramientas de widgets como Qt o GTK para su interfaz de usuario.

`rpm` verificará si esas dependencias están instaladas en su sistema y no podrá instalar el paquete si no lo están. En este caso, `rpm` listará lo que falta. Sin embargo, no puede resolver dependencias por sí mismo.

En el ejemplo a continuación, el usuario intentó instalar un paquete para el editor de imágenes GIMP, pero faltaban algunas dependencias:

```
# rpm -i gimp-2.8.22-1.el7.x86_64.rpm
error: Failed dependencies:
    babl(x86-64) >= 0.1.10 is needed by gimp-2:2.8.22-1.el7.x86_64
    gegl(x86-64) >= 0.2.0 is needed by gimp-2:2.8.22-1.el7.x86_64
    gimp-libs(x86-64) = 2:2.8.22-1.el7 is needed by gimp-2:2.8.22-1.el7.x86_64
    libbabl-0.1.so.0()(64bit) is needed by gimp-2:2.8.22-1.el7.x86_64
    libgegl-0.2.so.0()(64bit) is needed by gimp-2:2.8.22-1.el7.x86_64
    libgimp-2.0.so.0()(64bit) is needed by gimp-2:2.8.22-1.el7.x86_64
    libgimpbase-2.0.so.0()(64bit) is needed by gimp-2:2.8.22-1.el7.x86_64
    libgimpcolor-2.0.so.0()(64bit) is needed by gimp-2:2.8.22-1.el7.x86_64
    libgimpconfig-2.0.so.0()(64bit) is needed by gimp-2:2.8.22-1.el7.x86_64
    libgimpmath-2.0.so.0()(64bit) is needed by gimp-2:2.8.22-1.el7.x86_64
    libgimpmodule-2.0.so.0()(64bit) is needed by gimp-2:2.8.22-1.el7.x86_64
    libgimpthumb-2.0.so.0()(64bit) is needed by gimp-2:2.8.22-1.el7.x86_64
    libgimpui-2.0.so.0()(64bit) is needed by gimp-2:2.8.22-1.el7.x86_64
    libgimpwidgets-2.0.so.0()(64bit) is needed by gimp-2:2.8.22-1.el7.x86_64
    libmng.so.1()(64bit) is needed by gimp-2:2.8.22-1.el7.x86_64
    libwmf-0.2.so.7()(64bit) is needed by gimp-2:2.8.22-1.el7.x86_64
    libwmflite-0.2.so.7()(64bit) is needed by gimp-2:2.8.22-1.el7.x86_64
```

Depende del usuario encontrar los paquetes `.rpm` con las dependencias correspondientes e instalarlos. Los administradores de paquetes como `yum`, `zypper` y `dnf` tienen herramientas que pueden indicar qué paquete proporciona un archivo específico. Esos serán discutidos más adelante en esta lección.

Listar paquetes instalados

Para obtener una lista de todos los paquetes instalados en su sistema, use el `rpm -qa` (piense en "query all").

```
# rpm -qa
selinux-policy-3.13.1-229.el7.noarch
pciutils-libs-3.5.1-3.el7.x86_64
redhat-menus-12.0.2-8.el7.noarch
grubby-8.28-25.el7.x86_64
hunspell-en-0.20121024-6.el7.noarch
dejavu-fonts-common-2.33-6.el7.noarch
xorg-x11-drv-dummy-0.3.7-1.el7.1.x86_64
libevdev-1.5.6-1.el7.x86_64
[...]
```

Obtener Información de Paquetes

Para obtener información sobre un paquete *instalado*, como su número de versión, arquitectura, fecha de instalación, empaquetador, resumen, etc., use `rpm` con los parámetros `-qi` (piense en “query info”), seguido de el nombre del paquete. Por ejemplo:

```
# rpm -qi unzip
Name        : unzip
Version     : 6.0
Release     : 19.el7
Architecture: x86_64
Install Date: Sun 25 Aug 2019 05:14:39 PM EDT
Group       : Applications/Archiving
Size        : 373986
License     : BSD
Signature   : RSA/SHA256, Wed 25 Apr 2018 07:50:02 AM EDT, Key ID 24c6a8a7f4a80eb5
Source RPM  : unzip-6.0-19.el7.src.rpm
Build Date  : Wed 11 Apr 2018 01:24:53 AM EDT
Build Host  : x86-01.bsys.centos.org
Relocations : (not relocatable)
Packager    : CentOS BuildSystem <http://bugs.centos.org>
Vendor      : CentOS
URL         : http://www.info-zip.org/UnZip.html
Summary     : A utility for unpacking zip files
Description :
The unzip utility is used to list, test, or extract files from a zip
archive. Zip archives are commonly found on MS-DOS systems. The zip
utility, included in the zip package, creates zip archives. Zip and
unzip are both compatible with archives created by PKWARE(R)'s PKZIP
for MS-DOS, but the programs' options and default behaviors do differ
in some respects.

Instale el paquete unzip si necesita enumerar, probar o extraer archivos de un
archivo zip.
```

Para obtener una lista de los archivos que están dentro de un paquete *instalado*, use los parámetros `-ql` (piense en “query list”) seguido del nombre del paquete:

```
# rpm -ql unzip
/usr/bin/funzip
/usr/bin/unzip
/usr/bin/unzipsfx
/usr/bin/zipgrep
/usr/bin/zipinfo
/usr/share/doc/unzip-6.0
/usr/share/doc/unzip-6.0/BUGS
/usr/share/doc/unzip-6.0/LICENSE
/usr/share/doc/unzip-6.0/README
/usr/share/man/man1/funzip.1.gz
/usr/share/man/man1/unzip.1.gz
/usr/share/man/man1/unzipsfx.1.gz
/usr/share/man/man1/zipgrep.1.gz
/usr/share/man/man1/zipinfo.1.gz
```

Si desea obtener información o una lista de archivos de un paquete que aún no se ha instalado, simplemente agregue el parámetro `-p` a los comandos anteriores, seguido del nombre del archivo RPM (`FILENAME`). Entonces, `rpm -qi PACKAGENAME` se convierte en `rpm -qip FILENAME`, y `rpm -ql PACKAGENAME` se convierte en `rpm -qlp FILENAME`, como se muestra a continuación.

```
# rpm -qip atom.x86_64.rpm
Name        : atom
Version     : 1.40.0
Release     : 0.1
Architecture: x86_64
Install Date: (not installed)
Group       : Unspecified
Size        : 570783704
License     : MIT
Signature   : (none)
Source RPM  : atom-1.40.0-0.1.src.rpm
Build Date  : sex 09 ago 2019 12:36:31 -03
Build Host  : b01bbeaf3a88
Relocations : /usr
URL         : https://atom.io/
Summary     : A hackable text editor for the 21st Century.
Description :
A hackable text editor for the 21st Century.
```

```
# rpm -qlp atom.x86_64.rpm
/usr/bin/apm
/usr/bin/atom
/usr/share/applications/atom.desktop
/usr/share/atom
/usr/share/atom/LICENSE
/usr/share/atom/LICENSES.chromium.html
/usr/share/atom/atom
/usr/share/atom/atom.png
/usr/share/atom/blink_image_resources_200_percent.pak
/usr/share/atom/content_resources_200_percent.pak
/usr/share/atom/content_shell.pak

(listing goes on)
```

Averiguar qué paquete posee un archivo específico

Para averiguar qué archivo posee un paquete instalado, use el `-qf` (piense en “query file”) seguido de la ruta completa al archivo:

```
# rpm -qf /usr/bin/unzip
unzip-6.0-19.el7.x86_64
```

En el ejemplo anterior, el archivo `/usr/bin/unzip` pertenece al paquete `unzip-6.0-19.el7.x86_64`.

YellowDog Updater Modificado (YUM)

`yum` se desarrolló originalmente como *Yellow Dog Updater* (YUP), una herramienta para la gestión de paquetes en la distribución de Yellow Dog Linux. Con el tiempo, evolucionó para administrar paquetes en otros sistemas basados en RPM, como Fedora, CentOS, Red Hat Enterprise Linux y Oracle Linux.

Funcionalmente, es similar a la utilidad `apt` en los sistemas basados en Debian, pudiendo buscar, instalar, actualizar y eliminar paquetes y manejar automáticamente las dependencias. `yum` se puede usar para instalar un solo paquete o para actualizar un sistema completo a la vez.

Buscar Paquetes

Para instalar un paquete, necesita saber su nombre. Para esto, puede realizar una búsqueda con `yum search PATTERN`, donde `PATTERN` es el nombre del paquete que está buscando. El resultado es una lista de paquetes cuyos nombres o resúmenes contienen el patrón de búsqueda especificado. Por ejemplo,

si necesita una utilidad para manejar archivos comprimidos de 7Zip (con la extensión `.7z`) puede usar:

```
# yum search 7zip
Loaded plugins: fastestmirror, langpacks
Loading mirror speeds from cached hostfile
 * base: mirror.ufscar.br
 * epel: mirror.globo.com
 * extras: mirror.ufscar.br
 * updates: mirror.ufscar.br
=========================== N/S matchyutr54ed: 7zip ============================
p7zip-plugins.x86_64 : Additional plugins for p7zip
p7zip.x86_64 : Very high compression ratio file archiver
p7zip-doc.noarch : Manual documentation and contrib directory
p7zip-gui.x86_64 : 7zG - 7-Zip GUI version

  Name and summary matches only, use "search all" for everything.
```

Instalar, Actualizar y Eliminar Paquetes

Para instalar un paquete usando `yum`, use el comando `yum install PACKAGENAME`, donde `PACKAGENAME` es el nombre del paquete. `yum` buscará el paquete y las dependencias correspondientes de un repositorio en línea e instalará todo en su sistema.

```
# yum install p7zip
Loaded plugins: fastestmirror, langpacks
Loading mirror speeds from cached hostfile
 * base: mirror.ufscar.br
 * epel: mirror.globo.com
 * extras: mirror.ufscar.br
 * updates: mirror.ufscar.br
Resolving Dependencies
--> Running transaction check
---> Package p7zip.x86_64 0:16.02-10.el7 will be installed
--> Finished Dependency Resolution

Dependencies Resolved

==========================================================================
 Package        Arch            Version                Repository     Size
==========================================================================
Installing:
 p7zip          x86_64          16.02-10.el7           epel          604 k

Transaction Summary
==========================================================================
Install  1 Package

Total download size: 604 k
Installed size: 1.7 M
Is this ok [y/d/N]:
```

Para actualizar un paquete instalado, use `yum update PACKAGENAME`, donde `PACKAGENAME` es el nombre del paquete que desea actualizar. Por ejemplo:

```
# yum update wget
Loaded plugins: fastestmirror, langpacks
Loading mirror speeds from cached hostfile
 * base: mirror.ufscar.br
 * epel: mirror.globo.com
 * extras: mirror.ufscar.br
 * updates: mirror.ufscar.br
Resolving Dependencies
--> Running transaction check
---> Package wget.x86_64 0:1.14-18.el7 will be updated
---> Package wget.x86_64 0:1.14-18.el7_6.1 will be an update
--> Finished Dependency Resolution

Dependencies Resolved

===========================================================================
 Package     Arch          Version                    Repository      Size
===========================================================================
Updating:
 wget        x86_64        1.14-18.el7_6.1            updates        547 k

Transaction Summary
===========================================================================
Upgrade  1 Package

Total download size: 547 k
Is this ok [y/d/N]:
```

Si omite el nombre de un paquete, puede actualizar cada paquete en el sistema si existen actualizaciones disponibles.

Para verificar si hay una actualización disponible para un paquete específico, use `yum check-update PACKAGENAME`. Como antes, si omite el nombre del paquete, `yum` buscará actualizaciones para cada paquete instalado en el sistema.

Para eliminar un paquete instalado, use `yum remove PACKAGENAME`, donde `PACKAGENAME` es el nombre del paquete que desea eliminar.

Encontrar qué paquete proporciona un archivo específico

En un ejemplo anterior mostramos un intento de instalar el editor de imágenes `gimp`, que falló debido a dependencias insatisfechas. Sin embargo, `rpm` muestra qué archivos faltan, pero no lista el nombre de los paquetes que los proporcionan.

Por ejemplo, una de las dependencias que faltaba era `libgimpui-2.0.so.0`. Para ver qué paquete lo proporciona, puede usar `yum whatprovides`, seguido del nombre del archivo que está buscando:

```
# yum whatprovides libgimpui-2.0.so.0
Loaded plugins: fastestmirror, langpacks
Loading mirror speeds from cached hostfile
 * base: mirror.ufscar.br
 * epel: mirror.globo.com
 * extras: mirror.ufscar.br
 * updates: mirror.ufscar.br
2:gimp-libs-2.8.22-1.el7.i686 : GIMP libraries
Repo        : base
Matched from:
Provides    : libgimpui-2.0.so.0
```

La respuesta es `gimp-libs-2.8.22-1.el7.i686`. Luego puede instalar el paquete con el comando `yum install gimp-libs`.

Esto también funciona para archivos que ya están en su sistema. Por ejemplo, si desea saber de dónde proviene el archivo `/etc/hosts`, puede usar:

```
# yum whatprovides /etc/hosts
Loaded plugins: fastestmirror, langpacks
Loading mirror speeds from cached hostfile
 * base: mirror.ufscar.br
 * epel: mirror.globo.com
 * extras: mirror.ufscar.br
 * updates: mirror.ufscar.br
setup-2.8.71-10.el7.noarch : A set of system configuration and setup files
Repo        : base
Matched from:
Filename    : /etc/hosts
```

La respuesta es `setup-2.8.71-10.el7.noarch`.

Obtener información sobre un paquete

Para obtener información sobre un paquete, como su versión, arquitectura, descripción, tamaño y más, use `yum info PACKAGENAME` donde `PACKAGENAME` es el nombre del paquete para el que desea información:

```
# yum info firefox
Last metadata expiration check: 0:24:16 ago on Sat 21 Sep 2019 02:39:43 PM -03.
Installed Packages
Name         : firefox
Version      : 69.0.1
Release      : 3.fc30
Architecture : x86_64
Size         : 268 M
Source       : firefox-69.0.1-3.fc30.src.rpm
Repository   : @System
From repo    : updates
Summary      : Mozilla Firefox Web browser
URL          : https://www.mozilla.org/firefox/
License      : MPLv1.1 or GPLv2+ or LGPLv2+
Description  : Mozilla Firefox is an open-source web browser, designed
             : for standards compliance, performance and portability.
```

Gestión de repositorios de software

Para `yum`, los "repos" se enumeran en el directorio `/etc/yum.repos.d/`. Cada repositorio está representado por un archivo `.repo`, como `CentOS-Base.repo`.

El usuario puede agregar repositorios adicionales agregando un archivo `.repo` en el directorio mencionado anteriormente, o al final de `/etc/yum.conf`. Sin embargo, la forma recomendada de agregar o administrar repositorios es con la herramienta `yum-config-manager`.

Para agregar un repositorio, use el parámetro `--add-repo`, seguido de la URL a un archivo `.repo`.

```
# yum-config-manager --add-repo https://rpms.remirepo.net/enterprise/remi.repo
Loaded plugins: fastestmirror, langpacks
adding repo from: https://rpms.remirepo.net/enterprise/remi.repo
grabbing file https://rpms.remirepo.net/enterprise/remi.repo to
/etc/yum.repos.d/remi.repo
repo saved to /etc/yum.repos.d/remi.repo
```

Para obtener una lista de todos los repositorios disponibles, use `yum repolist all`. Obtendrá una salida similar a esta:

```
# yum repolist all
Loaded plugins: fastestmirror, langpacks
Loading mirror speeds from cached hostfile
 * base: mirror.ufscar.br
 * epel: mirror.globo.com
 * extras: mirror.ufscar.br
 * updates: mirror.ufscar.br
repo id                       repo name                     status
updates/7/x86_64              CentOS-7 - Updates            enabled:  2,500
updates-source/7              CentOS-7 - Updates Sources    disabled
```

Los repositorios `disabled` serán ignorados al instalar o actualizar el software. Para habilitar o deshabilitar un repositorio, use la utilidad `yum-config-manager`, seguido de la identificación del repositorio.

En el resultado anterior, la identificación del repositorio se muestra en la primera columna (`repo id`) de cada línea. Utilice solo la parte anterior a la primera `/`, por lo que la identificación para el repositorio `CentOS-7 - Updates` es `updates`, y no `updates/7/x86_64`.

```
# yum-config-manager --disable updates
```

El comando anterior deshabilitará el repositorio `updates`. Para volver a habilitarlo use:

```
# yum-config-manager --enable updates
```

NOTE

Yum almacena los paquetes descargados y los metadatos asociados en un directorio de caché (generalmente `/var/cache/yum`). A medida que el sistema se actualiza y se instalan nuevos paquetes, esta caché puede ser bastante grande. Para limpiar la caché y recuperar espacio en el disco, puede usar el comando `yum clean`, seguido de qué limpiar. Los parámetros más útiles son `packages` (`yum clean packages`) para eliminar los paquetes descargados y `metadata` (`yum clean metadata`) para eliminar los metadatos asociados. Consulte la página del manual para `yum` (teclee `man yum`) para obtener más información.

DNF

`dnf` es la herramienta de administración de paquetes utilizada en Fedora, y es una bifurcación de `yum`. Como tal, muchos de los comandos y parámetros son similares. Esta sección le dará solo una descripción rápida de `dnf`.

Buscar paquetes

`dnf search PATTERN`, donde `PATTERN` es lo que está buscando. Por ejemplo, `dnf search unzip` mostrará todos los paquetes que contienen la palabra `unzip` en el nombre o la descripción.

Obtener información de un paquete

`dnf info PACKAGENAME`

Instalar paquetes

`dnf install PACKAGENAME`, donde `PACKAGENAME` es el nombre del paquete que desea instalar. Puede encontrar el nombre realizando una búsqueda.

Eliminar paquetes

`dnf remove PACKAGENAME`

Actualizar paquetes

`dnf upgrade PACKAGENAME` para actualizar solo un paquete. Omita el nombre del paquete para actualizar todos los paquetes en el sistema.

Encontrar qué paquete proporciona un archivo específico

`dnf provides FILENAME`

Obtener una lista de todos los paquetes instalados en el sistema

`dnf list --installed`

Listar el contenido de un paquete

`dnf repoquery -l PACKAGENAME`

NOTE `dnf` tiene un sistema de ayuda incorporado, que muestra más información (como parámetros adicionales) para cada comando. Para usarlo, escriba `dnf help` seguido del comando, como `dnf help install`.

Gestión de repositorios de software

Al igual que con `yum` y `zypper`, `dnf` funciona con repositorios de software (repos). Cada distribución tiene una lista de repositorios predeterminados, y los administradores pueden agregar o eliminar repositorios según sea necesario.

Para obtener una lista de todos los repositorios disponibles, use `dnf repolist`. Para listar solo los repositorios habilitados, agregue la opción `--enabled`, y para listar solo los repositorios deshabilitados, agregue la opción `--disabled`.

```
# dnf repolist
Last metadata expiration check: 0:20:09 ago on Sat 21 Sep 2019 02:39:43 PM -03.
repo id                    repo name                                       status
*fedora                    Fedora 30 - x86_64                              56,582
*fedora-modular            Fedora Modular 30 - x86_64                         135
*updates                   Fedora 30 - x86_64 - Updates                    12,774
*updates-modular           Fedora Modular 30 - x86_64 - Updates               145
```

Para agregar un repositorio, use `dnf config-manager --add-repo URL`, donde `URL` es la URL completa del repositorio. Para habilitar un repositorio, use `dnf config-manager --set-enabled REPO_ID`.

Del mismo modo, para deshabilitar un repositorio use `dnf config-manager --set-disabled REPO_ID`. En ambos casos, REPO_ID es la ID única para el repositorio, que puede obtener usando `dnf repolist`. Los repositorios agregados están habilitados por defecto.

Los repositorios se almacenan en archivos `.repo` en el directorio `/etc/yum.repos.d/`, con exactamente la misma sintaxis utilizada para `yum`.

Zypper

`zypper` es la herramienta de gestión de paquetes utilizada en SUSE Linux y OpenSUSE. En cuanto a las características, es similar a `apt` y `yum`, pudiendo instalar, actualizar y eliminar paquetes de un sistema, con resolución de dependencias automatizada.

Actualización de los Índices de Paquetes

Al igual que otras herramientas de administración de paquetes, `zypper` funciona con repositorios que contienen paquetes y metadatos. Estos metadatos deben actualizarse de vez en cuando, para que la utilidad conozca los últimos paquetes disponibles. Para hacer una actualización, simplemente utilice el siguiente commando:

```
# zypper refresh
Repository 'Non-OSS Repository' is up to date.
Repository 'Main Repository' is up to date.
Repository 'Main Update Repository' is up to date.
Repository 'Update Repository (Non-Oss)' is up to date.
All repositories have been refreshed.
```

`zypper` tiene una función de actualización automática que se puede habilitar por repositorio, lo que significa que algunos repositorios pueden actualizarse automáticamente antes de una consulta o

instalación del paquete, y otros pueden necesitar actualizarse manualmente. Aprenderá a controlar esta función en breve.

Buscar Paquetes

Para buscar un paquete, use el operador `search` (o `se`), seguido del nombre del paquete:

```
# zypper se gnumeric
Loading repository data...
Reading installed packages...

S | Name           | Summary                            | Type
--+----------------+------------------------------------+--------
  | gnumeric       | Spreadsheet Application            | package
  | gnumeric-devel | Spreadsheet Application            | package
  | gnumeric-doc   | Documentation files for Gnumeric   | package
  | gnumeric-lang  | Translations for package gnumeric  | package
```

El operador de búsqueda también se puede utilizar para obtener una lista de todos los paquetes instalados en el sistema. Para hacerlo, use el parámetro `-i` sin un nombre de paquete, como en `zypper se -i`.

Para ver si está instalado un paquete específico, agregue el nombre del paquete al comando anterior. Por ejemplo, el siguiente comando buscará entre los paquetes instalados cualquier contenido que contenga “firefox” en el nombre:

```
# zypper se -i firefox
Loading repository data...
Reading installed packages...

S | Name                               | Summary                    | Type
--+------------------------------------+----------------------------+--------
i | MozillaFirefox                     | Mozilla Firefox Web B->    | package
i | MozillaFirefox-branding-openSUSE   | openSUSE branding of ->    | package
i | MozillaFirefox-translations-common | Common translations f->    | package
```

Para buscar solo entre paquetes *no-instalados*, agregue el parámetro `-u` al operador `se`.

Instalar, Actualizar y Eliminar Paquetes

Para instalar un paquete de software, use el operador `install` (o `in`), seguido del nombre del

paquete. Al igual que:

```
# zypper in unrar
zypper in unrar
Loading repository data...
Reading installed packages...
Resolving package dependencies...

The following NEW package is going to be installed:
  unrar

1 new package to install.
Overall download size: 141.2 KiB. Already cached: 0 B. After the operation,
additional 301.6 KiB will be used.
Continue? [y/n/v/...? shows all options] (y): y
Retrieving package unrar-5.7.5-lp151.1.1.x86_64
                                     (1/1), 141.2 KiB (301.6 KiB unpacked)
Retrieving: unrar-5.7.5-lp151.1.1.x86_64.rpm .......................[done]
Checking for file conflicts: .......................................[done]
(1/1) Installing: unrar-5.7.5-lp151.1.1.x86_64 .....................[done]
```

`zypper` también se puede usar para instalar un paquete RPM en disco, mientras se intenta satisfacer sus dependencias usando paquetes de los repositorios. Para hacerlo, solo proporcione la ruta completa al paquete en lugar del nombre del paquete, como `zypper in /home/john/newpackage.rpm`.

Para actualizar los paquetes instalados en el sistema, use `zypper update`. Al igual que en el proceso de instalación, esto mostrará una lista de paquetes para instalar/actualizar antes de preguntar si desea continuar.

Si solo desea listar las actualizaciones disponibles, sin instalar nada, puede usar `zypper list-updates`.

Para eliminar un paquete, use el operador `remove` (o `rm`), seguido del nombre del paquete:

```
# zypper rm unrar
Loading repository data...
Reading installed packages...
Resolving package dependencies...

The following package is going to be REMOVED:
  unrar

1 package to remove.
After the operation, 301.6 KiB will be freed.
Continue? [y/n/v/...? shows all options] (y): y
(1/1) Removing unrar-5.7.5-lp151.1.1.x86_64 ........................[done]
```

Tenga en cuenta que eliminar un paquete también elimina cualquier otro paquete que dependa de él. Por ejemplo:

```
# zypper rm libgimp-2_0-0
Loading repository data...
Warning: No repositories defined. Operating only with the installed resolvables.
Nothing can be installed.
Reading installed packages...
Resolving package dependencies...

The following 6 packages are going to be REMOVED:
  gimp gimp-help gimp-lang gimp-plugins-python libgimp-2_0-0
  libgimpui-2_0-0

6 packages to remove.
After the operation, 98.0 MiB will be freed.
Continue? [y/n/v/...? shows all options] (y):
```

Encontrar qué paquete proporciona un archivo específico

Para ver qué paquetes contiene un archivo específico, use el operador de búsqueda seguido del parámetro `--provides` y el nombre del archivo (o la ruta completa). Por ejemplo, si desea saber qué paquetes contienen el archivo `libgimpmodule-2.0.so.0` en `/usr/lib64/` use:

```
# zypper se --provides /usr/lib64/libgimpmodule-2.0.so.0
Loading repository data...
Reading installed packages...

S | Name          | Summary                                        | Type
--+---------------+------------------------------------------------+--------
i | libgimp-2_0-0 | The GNU Image Manipulation Program - Libra-> | package
```

Obtener Información de Paquetes

Para ver los metadatos asociados con un paquete, use el operador `info` seguido del nombre del paquete. Esto le proporcionará el repositorio de origen, el nombre del paquete, la versión, la arquitectura, el proveedor, el tamaño instalado, si está instalado o no, el estado (si está actualizado), el paquete fuente y una descripción.

```
# zypper info gimp
Loading repository data...
Reading installed packages...

Information for package gimp:
 -----------------------------
Repository     : Main Repository
Name           : gimp
Version        : 2.8.22-lp151.4.6
Arch           : x86_64
Vendor         : openSUSE
Installed Size : 29.1 MiB
Installed      : Yes (automatically)
Status         : up-to-date
Source package : gimp-2.8.22-lp151.4.6.src
Summary        : The GNU Image Manipulation Program
Description    :
    The GIMP is an image composition and editing program, which can be
    used for creating logos and other graphics for Web pages. The GIMP
    offers many tools and filters, and provides a large image
    manipulation toolbox, including channel operations and layers,
    effects, subpixel imaging and antialiasing, and conversions, together
    with multilevel undo. The GIMP offers a scripting facility, but many
    of the included scripts rely on fonts that we cannot distribute.
```

Gestión de repositorios de software

`zypper` también se puede usar para administrar repositorios de software. Para ver una lista de todos los repositorios actualmente registrados en su sistema, use `zypper repos`:

```
# zypper repos
Repository priorities are without effect. All enabled repositories share the same
priority.

#  | Alias                     | Name                               | Enabled | GPG
Check | Refresh
---+---------------------------+------------------------------------+---------+
-----------+--------
 1 | openSUSE-Leap-15.1-1      | openSUSE-Leap-15.1-1               | No      |
----      | ----
 2 | repo-debug                | Debug Repository                   | No      |
----      | ----
 3 | repo-debug-non-oss        | Debug Repository (Non-OSS)         | No      |
----      | ----
 4 | repo-debug-update         | Update Repository (Debug)          | No      |
----      | ----
 5 | repo-debug-update-non-oss | Update Repository (Debug, Non-OSS) | No      |
----      | ----
 6 | repo-non-oss              | Non-OSS Repository                 | Yes     | (r
) Yes   | Yes
 7 | repo-oss                  | Main Repository                    | Yes     | (r
) Yes   | Yes
 8 | repo-source               | Source Repository                  | No      |
----      | ----
 9 | repo-source-non-oss       | Source Repository (Non-OSS)        | No      |
----      | ----
10 | repo-update               | Main Update Repository             | Yes     | (r
) Yes   | Yes
11 | repo-update-non-oss       | Update Repository (Non-Oss)        | Yes     | (r
) Yes   | Yes
```

En la columna `Enabled` se puede verificar que algunos repositorios están habilitados, mientras que otros no. Puede cambiar esto con el operador `modifyrepo`, seguido del parámetro `-e` (habilitar) o `-d` (deshabilitar) y el alias del repositorio (la segunda columna en la salida anterior).

```
# zypper modifyrepo -d repo-non-oss
Repository 'repo-non-oss' has been successfully disabled.

# zypper modifyrepo -e repo-non-oss
Repository 'repo-non-oss' has been successfully enabled.
```

Anteriormente mencionamos que `zypper` tiene la capacidad de actualización automática que se puede habilitar por repositorio. Cuando está habilitado, este indicador hará que `zypper` ejecute una operación de actualización (lo mismo que ejecutar `zypper refresh`) antes de trabajar con el repositorio especificado. Esto se puede controlar con los parámetros `-f` y `-F` del operador `modifyrepo`:

```
# zypper modifyrepo -F repo-non-oss
Autorefresh has been disabled for repository 'repo-non-oss'.

# zypper modifyrepo -f repo-non-oss
Autorefresh has been enabled for repository 'repo-non-oss'.
```

Agregar y quitar repositorios

Para agregar un nuevo repositorio de software para `zypper`, use el operador `addrepo` seguido de la URL del repositorio y el nombre del repositorio, como se muestra a continuación:

```
# zypper addrepo http://packman.inode.at/suse/openSUSE_Leap_15.1/ packman
Adding repository 'packman' .......................................[done]
Repository 'packman' successfully added

URI         : http://packman.inode.at/suse/openSUSE_Leap_15.1/
Enabled     : Yes
GPG Check   : Yes
Autorefresh : No
Priority    : 99 (default priority)

Repository priorities are without effect. All enabled repositories share the same
priority.
```

Al agregar un repositorio, puede habilitar las actualizaciones automáticas con el parámetro `-f`. Los repositorios agregados están habilitados de manera predeterminada, pero puede agregar y deshabilitar un repositorio al mismo tiempo utilizando el parámetro `-d`.

Para eliminar un repositorio, use el operador `removerepo`, seguido del nombre del repositorio (Alias). Para eliminar el repositorio agregado en el ejemplo anterior, el comando sería:

```
# zypper removerepo packman
Removing repository 'packman' ......................................[done]
Repository 'packman' has been removed.
```

Ejercicios Guiados

1. Usando `rpm` en un sistema Red Hat Enterprise Linux, ¿cómo instalaría el paquete `file-roller-3.28.1-2.el7.x86_64.rpm` mostrando una barra de progreso durante la instalación?

2. Usando `rpm`, descubra qué paquete contiene el archivo `/etc/redhat-release`.

3. ¿Cómo usaría `yum` para buscar actualizaciones para todos los paquetes en el sistema?

4. Usando `zypper`, ¿cómo deshabilitaría un repositorio llamado repo-extras?

5. Si tiene un archivo `.repo` que describe un nuevo repositorio, ¿dónde se debe colocar este archivo para que DNF lo reconozca?

Ejercicios Exploratorios

1. ¿Cómo usaría `zypper` para averiguar qué paquete posee el archivo `/usr/sbin/swapon`?

2. ¿Cómo puede obtener una lista de todos los paquetes instalados en el sistema usando `dnf`?

3. Usando `dnf`, ¿cuál es el comando para agregar un repositorio ubicado en `https://www.example.url/home:reponame.repo` al sistema?

4. ¿Cómo puede usar `zypper` para verificar si el paquete `unzip` está instalado?

5. Usando `yum`, descubra qué paquete proporciona el archivo `/bin/wget`.

Resumen

En esta lección aprendimos:

- ¿Cómo usar `rpm` para instalar, actualizar y eliminar paquetes?
- ¿Cómo usar `yum`, `zypper` y `dnf`?
- ¿Cómo obtener información sobre un paquete?
- ¿Cómo obtener una lista de los contenidos de un paquete?
- ¿Cómo averiguar de qué paquete proviene un archivo?
- ¿Cómo listar, agregar, eliminar, habilitar o deshabilitar repositorios de software?

Los siguientes comandos se discutieron en esta lección:

- `rpm`
- `yum`
- `dnf`
- `zypper`

Respuestas a los ejercicios guiados

1. Usando `rpm` en un sistema Red Hat Enterprise Linux, ¿cómo instalaría el paquete `file-roller-3.28.1-2.el7.x86_64.rpm` mostrando una barra de progreso durante la instalación?

   Use el parámetro `-i` para instalar un paquete y la opción `-h` para habilitar las “hash marks” que muestran el progreso de la instalación. Entonces, la respuesta es: `rpm -ih file-roller-3.28.1-2.el7.x86_64.rpm`.

2. Usando `rpm`, descubra qué paquete contiene el archivo `/etc/redhat-release`.

   Para consultar información sobre un archivo use el parámetro `-qf`: `rpm -qf /etc/redhat-release`.

3. ¿Cómo usaría `yum` para buscar actualizaciones para todos los paquetes en el sistema?

   Utilice la operación `check-update` *sin* un nombre de paquete: `yum check-update`.

4. Usando `zypper`, ¿cómo deshabilitaría un repositorio llamado repo-extras?

   Use la operación `modifyrepo` para cambiar los parámetros de un repositorio, y el parámetro `-d` para deshabilitarlo: `zypper modifyrepo -d repo-extras`.

5. Si tiene un archivo `.repo` que describe un nuevo repositorio, ¿dónde se debe colocar este archivo para que DNF lo reconozca?

   Los archivos `.repo` para DNF deben colocarse en el mismo lugar utilizado por YUM, dentro de `/etc/yum.repos.d/`.

Respuestas a ejercicios exploratorios

1. ¿Cómo usarías `zypper` para averiguar qué paquete posee el archivo `/usr/sbin/swapon`?

   Utilice el operador `se` (search) y el parámetro `--provides`: `zypper se --provides /usr/sbin/swapon`.

2. ¿Cómo puede obtener una lista de todos los paquetes instalados en el sistema usando `dnf`?

   Utilice el operador `list`, seguido del parámetro `--installed`: `dnf list --installed`.

3. Usando `dnf`, ¿cuál es el comando para agregar un repositorio ubicado en `https://www.example.url/home:reponame.repo` al sistema?

   Trabajar con repositorios es un “configuration change”, así que utilice el parámetro `config-manager` y el parámetro `--add_repo`: `dnf config-manager --add_repo https://www.example.url/home:reponame.repo`.

4. ¿Cómo puede usar 'zypper' para verificar si el paquete 'unzip' está instalado?

   Debe hacer una búsqueda (`se`) en los paquetes instalados (`-i`): `zypper se -i unzip`.

5. Usando `yum`, descubra qué paquete proporciona el archivo `/bin/wget`.

   Para averiguar qué paquete proporciona un archivo, use `whatprovides` y el nombre del archivo: `yum whatprovides /bin/wget`.



102.6 Linux como sistema virtualizado

Referencia al objetivo del LPI

LPIC-1, Exam 101, Objective 102.6

Importancia

1

Áreas de conocimiento clave

- Entender el concepto general de máquina virtual y contenedor
- Entender elementos comunes de una máquina virtual en un entorno de nube de tipo IaaS, tales como instancia de computación, almacenamiento de bloques y redes
- Entender las propiedades únicas de un sistema Linux que tienen que cambiar cuando el sistema se clona o se usa como plantilla
- Entender cómo se usan las imágenes de sistema para desplegar máquinas virtuales, instancias de nube y contenedores
- Entender las extensiones de Linux que permiten la integración con un producto de virtualización
- Conocimientos de cloud-init

Lista parcial de archivos, términos y utilidades

- Máquina virtual
- Contenedor Linux
- Contenedor de aplicaciones
- Controladores de la máquina huésped
- Claves de host SSH

- ID de D-Bus



102.6 Lección 1

Certificación:	LPIC-1
Versión:	5.0
Tema:	102 Instalación de Linux y Administración de Paquetes
Objectivo:	102.6 Linux como huésped de virtualización
Lección:	1 de 1

Introducción

Una de las grandes fortalezas de Linux es su versatilidad. Un aspecto de esta versatilidad es la capacidad de usar Linux como medio para alojar otros sistemas operativos, o aplicaciones individuales, en un entorno completamente aislado y seguro. Esta lección se centrará en los conceptos de virtualización y tecnologías de contenedor, junto con algunos detalles técnicos que deben tenerse en cuenta al implementar una máquina virtual en una plataforma en la nube.

Descripción general de virtualización

La virtualización es una tecnología que permite que una plataforma de software, llamada hipervisor (*hypervisor*), ejecute procesos que contienen un sistema informático completamente emulado. El hipervisor es responsable de administrar los recursos del hardware físico que pueden ser utilizados por máquinas virtuales individuales. Estas máquinas virtuales se denominan *guests* del hipervisor. Una máquina virtual tiene muchos aspectos de una computadora física emulada en software, como el BIOS del sistema y los controladores de disco del disco duro. Una máquina virtual a menudo usará imágenes de disco duro que se almacenan como archivos individuales, y tendrá acceso a la RAM y CPU de la máquina host a través del software del hipervisor. El hipervisor separa el acceso a los

recursos de hardware del sistema host entre los guest, lo que permite que múltiples sistemas operativos se ejecuten en un solo sistema host.

Los hipervisores de uso común en Linux son:

Xen

Xen es un hipervisor de código abierto de tipo 1, lo que significa que no depende de un sistema operativo subyacente para funcionar. Un hipervisor de este tipo se conoce como un hipervisor de *bare-metal hypervisor*, ya que la computadora puede arrancar directamente en el hipervisor.

KVM

Kernel Virtual Machine es un módulo de kernel de Linux para virtualización. KVM es un hipervisor tanto del hipervisor Tipo 1 como del Tipo 2 porque, aunque necesita un sistema operativo Linux genérico para funcionar, puede funcionar perfectamente como hipervisor al integrarse con una instalación Linux en ejecución. Las máquinas virtuales implementadas con KVM usan el demonio `libvirt` y las utilidades de software asociadas para ser creadas y administradas.

VirtualBox

Una aplicación de escritorio popular que facilita la creación y administración de máquinas virtuales. Oracle VM VirtualBox es multiplataforma y funciona en Linux, macOS y Microsoft Windows. Como VirtualBox requiere de un sistema operativo subyacente para ejecutarse, es un hipervisor de tipo 2.

Algunos hipervisores permiten la reubicación dinámica de una máquina virtual. El proceso de mover una máquina virtual de una instalación de hipervisor a otra se llama *migration*, y las técnicas involucradas difieren entre las implementaciones de hipervisor. Algunas migraciones solo se pueden realizar cuando el sistema invitado se ha apagado por completo, y otras se pueden realizar mientras el invitado se está ejecutando (denominado *live migration*). Dichas técnicas pueden ser útiles durante los mantenimientos de los hipervisores, o para la resistencia del sistema cuando un hipervisor deja de funcionar y el huésped puede ser trasladado a un hipervisor que esté funcionando.

Tipos de Máquinas Virtuales

Hay tres tipos principales de máquinas virtuales: el *fully virtualized* guest (un invitado totalmente virtualizado), el *paravirtualized* guest (un invitado paravirtualizado) y el *hybrid* guest (un invitado híbrido).

Totalmente virtualizado (Fully Virtualized)

Todas las instrucciones que se espera que ejecute un sistema operativo invitado deben poder ejecutarse dentro de una instalación de sistema operativo totalmente virtualizada. La razón de

esto es que no se instalan controladores de software adicionales dentro del huésped para traducir las instrucciones a hardware simulado o real. Un invitado totalmente virtualizado es aquel en el que el invitado (o HardwareVM) desconoce que es una instancia de máquina virtual en ejecución. Para que este tipo de virtualización tenga lugar en hardware basado en x86, las extensiones de CPU Intel VT-x o AMD-V deben estar habilitadas en el sistema que tiene instalado el hipervisor. Esto se puede hacer desde un menú de configuración de firmware BIOS o UEFI.

Paravirtualizado (Paravirtualized)

Un invitado paravirtualizado (o PVM) es aquel en el que el sistema operativo invitado es consciente de que es una instancia de máquina virtual en ejecución. Este tipo de invitados utilizará un kernel modificado y controladores especiales (conocidos como "controladores invitados") que ayudarán al sistema operativo invitado a utilizar los recursos de software y hardware del hipervisor. El rendimiento de un huésped paravirtualizado es a menudo mejor que el del huésped totalmente virtualizado debido a la ventaja que proporcionan estos controladores de software.

Híbrido (Hybrid)

La paravirtualización y la virtualización completa se pueden combinar para permitir que los sistemas operativos no modificados reciban un rendimiento de E/S casi nativo mediante el uso de controladores paravirtualizados en sistemas operativos completamente virtualizados. Los controladores paravirtualizados contienen controladores de almacenamiento y dispositivos de red con disco mejorado y rendimiento de E/S de red.

Las plataformas de virtualización a menudo proporcionan controladores invitados empaquetados para sistemas operativos virtualizados. El KVM utiliza controladores del proyecto *Virtio*, mientras que Oracle VM VirtualBox utiliza *Guest Extensions* disponibles desde un archivo de imagen de CD-ROM ISO descargable.

Ejemplo de máquina virtual `libvirt`

Veremos un ejemplo de máquina virtual que es administrada por `libvirt` y usa el hipervisor KVM. Una máquina virtual a menudo consiste en un grupo de archivos, principalmente un archivo XML que define la máquina virtual (como su configuración de hardware, conectividad de red, capacidades de visualización y más) y un archivo de imagen de disco duro asociado que contiene la instalación del sistema operativo y su software.

Primero, comencemos a examinar un archivo de configuración XML de ejemplo para una máquina virtual y su entorno de red:

```
$ ls /etc/libvirt/qemu
total 24
drwxr-xr-x 3 root root 4096 Oct 29 17:48 networks
-rw------- 1 root root 5667 Jun 29 17:17 rhel8.0.xml
```

NOTE La parte `qemu` de la ruta del directorio se refiere al software subyacente en el que confían las máquinas virtuales basadas en KVM. El proyecto QEMU proporciona software para que el hipervisor emule dispositivos de hardware que usará la máquina virtual, como controladores de disco, acceso a la CPU del host, emulación de tarjeta de red y más.

Tenga en cuenta que hay un directorio llamado `networks`. Este directorio contiene archivos de definición (también usando XML) que crean configuraciones de red que las máquinas virtuales pueden usar. Este hipervisor solo utiliza una red, por lo que solo hay un archivo de definición que contiene una configuración para un segmento de red virtual que utilizarán estos sistemas.

```
$ ls -l /etc/libvirt/qemu/networks/
total 8
drwxr-xr-x 2 root root 4096 Jun 29 17:15 autostart
-rw------- 1 root root  576 Jun 28 16:39 default.xml
$ sudo cat /etc/libvirt/qemu/networks/default.xml
<!--
WARNING: THIS IS AN AUTO-GENERATED FILE. CHANGES TO IT ARE LIKELY TO BE
OVERWRITTEN AND LOST. Changes to this xml configuration should be made using:
  virsh net-edit default
or other application using the libvirt API.
-->

<network>
  <name>default</name>
  <uuid>55ab064f-62f8-49d3-8d25-8ef36a524344</uuid>
  <forward mode='nat'/>
  <bridge name='virbr0' stp='on' delay='0'/>
  <mac address='52:54:00:b8:e0:15'/>
  <ip address='192.168.122.1' netmask='255.255.255.0'>
    <dhcp>
      <range start='192.168.122.2' end='192.168.122.254'/>
    </dhcp>
  </ip>
</network>
```

Esta definición incluye una red privada de Clase C y un dispositivo de hardware emulado para actuar como enrutador para esta red. También hay un rango de direcciones IP para que el hipervisor las use con una implementación de servidor DHCP que puede asignarse a las máquinas virtuales que usan esta red. Esta configuración de red también utiliza la traducción de direcciones de red (NAT) para reenviar paquetes a otras redes, como la LAN del hipervisor.

Ahora dirigimos nuestra atención a un archivo de definición de máquina virtual Red Hat Enterprise Linux 8. (las secciones de nota especial están en negrita):

```
$ sudo cat /etc/libvirt/qemu/rhel8.0.xml
<!--
WARNING: THIS IS AN AUTO-GENERATED FILE. CHANGES TO IT ARE LIKELY TO BE
OVERWRITTEN AND LOST. Changes to this xml configuration should be made using:
  virsh edit rhel8.0
or other application using the libvirt API.
-->

<domain type='kvm'>
  <name>rhel8.0</name>
  <uuid>fadd8c5d-c5e1-410e-b425-30da7598d0f6</uuid>
  <metadata>
    <libosinfo:libosinfo
xmlns:libosinfo="http://libosinfo.org/xmlns/libvirt/domain/1.0">
      <libosinfo:os id="http://redhat.com/rhel/8.0"/>
    </libosinfo:libosinfo>
  </metadata>
  <memory unit='KiB'>4194304</memory>
  <currentMemory unit='KiB'>4194304</currentMemory>
  <vcpu placement='static'>2</vcpu>
  <os>
    <type arch='x86_64' machine='pc-q35-3.1'>hvm</type>
    <boot dev='hd'/>
  </os>
  <features>
    <acpi/>
    <apic/>
    <vmport state='off'/>
  </features>
  <cpu mode='host-model' check='partial'>
    <model fallback='allow'/>
  </cpu>
  <clock offset='utc'>
    <timer name='rtc' tickpolicy='catchup'/>
    <timer name='pit' tickpolicy='delay'/>
```

```
    <timer name='hpet' present='no'/>
  </clock>
  <on_poweroff>destroy</on_poweroff>
  <on_reboot>restart</on_reboot>
  <on_crash>destroy</on_crash>
  <pm>
    <suspend-to-mem enabled='no'/>
    <suspend-to-disk enabled='no'/>
  </pm>
  <devices>
    <emulator>/usr/bin/qemu-system-x86_64</emulator>
    <disk type='file' device='disk'>
      <driver name='qemu' type='qcow2'/>
      <source file='/var/lib/libvirt/images/rhel8'/>
      <target dev='vda' bus='virtio'/>
      <address type='pci' domain='0x0000' bus='0x04' slot='0x00' function='0x0'/>
    </disk>
    <controller type='usb' index='0' model='qemu-xhci' ports='15'>
      <address type='pci' domain='0x0000' bus='0x02' slot='0x00' function='0x0'/>
    </controller>
    <controller type='sata' index='0'>
      <address type='pci' domain='0x0000' bus='0x00' slot='0x1f' function='0x2'/>
    </controller>
    <controller type='pci' index='0' model='pcie-root'/>
    <controller type='pci' index='1' model='pcie-root-port'>
      <model name='pcie-root-port'/>
      <target chassis='1' port='0x10'/>
      <address type='pci' domain='0x0000' bus='0x00' slot='0x02' function='0x0'
multifunction='on'/>
    </controller>
    <controller type='pci' index='2' model='pcie-root-port'>
      <model name='pcie-root-port'/>
      <target chassis='2' port='0x11'/>
      <address type='pci' domain='0x0000' bus='0x00' slot='0x02' function='0x1'/>
    </controller>
    <controller type='pci' index='3' model='pcie-root-port'>
      <model name='pcie-root-port'/>
      <target chassis='3' port='0x12'/>
      <address type='pci' domain='0x0000' bus='0x00' slot='0x02' function='0x2'/>
    </controller>
    <controller type='pci' index='4' model='pcie-root-port'>
      <model name='pcie-root-port'/>
      <target chassis='4' port='0x13'/>
      <address type='pci' domain='0x0000' bus='0x00' slot='0x02' function='0x3'/>
    </controller>
```

```
    <controller type='pci' index='5' model='pcie-root-port'>
      <model name='pcie-root-port'/>
      <target chassis='5' port='0x14'/>
      <address type='pci' domain='0x0000' bus='0x00' slot='0x02' function='0x4'/>
    </controller>
    <controller type='pci' index='6' model='pcie-root-port'>
      <model name='pcie-root-port'/>
      <target chassis='6' port='0x15'/>
      <address type='pci' domain='0x0000' bus='0x00' slot='0x02' function='0x5'/>
    </controller>
    <controller type='pci' index='7' model='pcie-root-port'>
      <model name='pcie-root-port'/>
      <target chassis='7' port='0x16'/>
      <address type='pci' domain='0x0000' bus='0x00' slot='0x02' function='0x6'/>
    </controller>
    <controller type='virtio-serial' index='0'>
      <address type='pci' domain='0x0000' bus='0x03' slot='0x00' function='0x0'/>
    </controller>
    <interface type='network'>
      <mac address='52:54:00:50:a7:18'/>
      <source network='default'/>
      <model type='virtio'/>
      <address type='pci' domain='0x0000' bus='0x01' slot='0x00' function='0x0'/>
    </interface>
    <serial type='pty'>
      <target type='isa-serial' port='0'>
        <model name='isa-serial'/>
      </target>
    </serial>
    <console type='pty'>
      <target type='serial' port='0'/>
    </console>
    <channel type='unix'>
      <target type='virtio' name='org.qemu.guest_agent.0'/>
      <address type='virtio-serial' controller='0' bus='0' port='1'/>
    </channel>
    <channel type='spicevmc'>
      <target type='virtio' name='com.redhat.spice.0'/>
      <address type='virtio-serial' controller='0' bus='0' port='2'/>
    </channel>
    <input type='tablet' bus='usb'>
      <address type='usb' bus='0' port='1'/>
    </input>
    <input type='mouse' bus='ps2'/>
    <input type='keyboard' bus='ps2'/>
```

```
    <graphics type='spice' autoport='yes'>
      <listen type='address'/>
      <image compression='off'/>
    </graphics>
    <sound model='ich9'>
      <address type='pci' domain='0x0000' bus='0x00' slot='0x1b' function='0x0'/>
    </sound>
    <video>
      <model type='virtio' heads='1' primary='yes'/>
      <address type='pci' domain='0x0000' bus='0x00' slot='0x01' function='0x0'/>
    </video>
    <redirdev bus='usb' type='spicevmc'>
      <address type='usb' bus='0' port='2'/>
    </redirdev>
    <redirdev bus='usb' type='spicevmc'>
      <address type='usb' bus='0' port='3'/>
    </redirdev>
    <memballoon model='virtio'>
      <address type='pci' domain='0x0000' bus='0x05' slot='0x00' function='0x0'/>
    </memballoon>
    <rng model='virtio'>
      <backend model='random'>/dev/urandom</backend>
      <address type='pci' domain='0x0000' bus='0x06' slot='0x00' function='0x0'/>
    </rng>
  </devices>
</domain>
```

Este archivo define una serie de configuraciones de hardware que utiliza el sistema invitado (*guest*), como la cantidad de RAM que le habrá asignado, el número de núcleos de CPU del hipervisor al que tendrá acceso el invitado, el archivo de imagen del disco duro que está asociado (bajo la etiqueta `disk`), sus capacidades de visualización (a través del protocolo SPICE) y el acceso del invitado a dispositivos USB, así como a la entrada emulada de teclado y mouse.

Ejemplo de almacenamiento en disco de una máquina virtual

La imagen del disco duro de esta máquina virtual reside en `/var/lib/libvirt/images/rhel8`. Aquí está la imagen de disco en este hipervisor:

```
$ sudo ls -lh /var/lib/libvirt/images/rhel8
-rw------- 1 root root 5.5G Oct 25 15:57 /var/lib/libvirt/images/rhel8
```

El tamaño actual de esta imagen de disco ocupa solo 5,5 GB de espacio en el hipervisor. Sin embargo,

el sistema operativo dentro de este invitado ve un disco de 23,3 GB de tamaño, como lo demuestra la salida del siguiente comando desde la máquina virtual en ejecución:

```
$ lsblk
NAME            MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
vda             252:0    0 23.3G  0 disk
├─vda1          252:1    0    1G  0 part /boot
└─vda2          252:2    0 22.3G  0 part
  ├─rhel-root 253:0    0   20G  0 lvm  /
  └─rhel-swap 253:1    0  2.3G  0 lvm  [SWAP]
```

Esto se debe al tipo de aprovisionamiento de disco utilizado para este invitado. Hay varios tipos de imágenes de disco que una máquina virtual puede usar, pero los dos tipos principales son:

COW

Copy-on-write (también conocido como *thin-provisioning* o *sparse images*) es un método en el que se crea un archivo de disco con un límite de tamaño superior predefinido. El tamaño de la imagen del disco solo aumenta a medida que se escriben nuevos datos en el disco. Al igual que en el ejemplo anterior, el sistema operativo invitado ve el límite de disco predefinido de 23,3 GB, pero solo ha escrito 5,5 GB de datos en el archivo del disco. El formato de imagen de disco utilizado para la máquina virtual de ejemplo es `qcow2`, que es un formato de archivo QEMU COW.

RAW

Un tipo de disco *raw* o *full* es un archivo que tiene todo su espacio preasignado. Por ejemplo, un archivo de imagen de disco sin formato de 10 GB consume 10 GB de espacio real en el hipervisor. Hay un beneficio de rendimiento para este estilo de disco, ya que todo el espacio en disco necesario ya existe, por lo que el hipervisor subyacente puede simplemente escribir datos en el disco sin el impacto del rendimiento de monitorear la imagen del disco para asegurarse de que aún no ha alcanzado su límite y extender el tamaño del archivo a medida que se escriben nuevos datos.

Existen otras plataformas de administración de virtualización como *Red Hat Enterprise Virtualization* y *oVirt* que pueden usar discos físicos para actuar como ubicaciones de almacenamiento de respaldo para el sistema operativo de una máquina virtual. Estos sistemas pueden utilizar la red de área de almacenamiento (SAN) o dispositivos de almacenamiento conectados a la red (NAS) para escribir sus datos, y el hipervisor realiza un seguimiento de qué ubicaciones de almacenamiento pertenecen a qué máquinas virtuales. Estos sistemas de almacenamiento pueden usar tecnologías como la administración de volumen lógico (LVM) para aumentar o reducir el tamaño del almacenamiento en disco de una máquina virtual según sea necesario, y para ayudar en la creación y administración de instantáneas de almacenamiento.

Trabajando con plantillas de máquinas virtuales

Dado que las máquinas virtuales generalmente son solo archivos que se ejecutan en un hipervisor, es fácil crear plantillas que se pueden personalizar para escenarios de implementación particulares. A menudo, una máquina virtual tendrá una instalación básica del sistema operativo y algunos ajustes de configuración de autenticación preconfigurados para facilitar futuros lanzamientos del sistema. Esto reduce la cantidad de tiempo que lleva construir un nuevo sistema al reducir la cantidad de trabajo que a menudo se repite, como la instalación de paquetes base y la configuración regional.

Esta plantilla de máquina virtual podría copiarse luego a un nuevo sistema invitado (*guest*). En este caso, se cambiaría el nombre del nuevo invitado (*guest*), se generaría una nueva dirección MAC para su interfaz de red y se podrían realizar otras modificaciones dependiendo de su uso previsto.

El D-Bus Machine ID

Muchas instalaciones de Linux utilizarán un número de identificación de máquina generado en el momento de la instalación, llamado *D-Bus machine ID*. Sin embargo, si una máquina virtual se *clona* para ser utilizada como plantilla para otras instalaciones de máquinas virtuales, se necesitaría crear una nueva ID de máquina D-Bus para garantizar que los recursos del sistema del hipervisor se dirijan al sistema invitado (*guest*) apropiadamente.

El siguiente comando se puede usar para validar que existe una ID de máquina D-Bus para el sistema en ejecución:

```
$ dbus-uuidgen --ensure
```

Si no se muestran mensajes de error, existe una ID para el sistema. Para ver la ID actual de la máquina D-Bus, ejecute lo siguiente:

```
$ dbus-uuidgen --get
17f2e0698e844e31b12ccd3f9aa4d94a
```

La cadena de texto que se muestra es el número de identificación actual. No hay dos sistemas Linux que se ejecuten en un hipervisor que tengan la misma ID de máquina D-Bus.

La ID de la máquina D-Bus se encuentra en `/var/lib/dbus/machine-id` y está simbólicamente vinculada a `/etc/machine-id`. Se desaconseja cambiar este número de identificación en un sistema en ejecución, ya que es probable que ocurran inestabilidades y fallas del sistema. Si dos máquinas virtuales tienen la misma ID de máquina D-Bus, siga el procedimiento a continuación para generar una nueva:


```
$ sudo rm -f /etc/machine-id
$ sudo dbus-uuidgen --ensure=/etc/machine-id
```

En el caso de que `/var/lib/dbus/machine-id` no sea un enlace simbólico a `/etc/machine-id`, entonces será necesario eliminar `/var/lib/dbus/machine-id`.

Implementación de máquinas virtuales en la nube

Hay una multitud de proveedores de IaaS (*infrastructure as a service*) disponibles que ejecutan sistemas de hipervisor y que pueden implementar imágenes virtuales de invitados (*guest*) para una organización. Prácticamente todos estos proveedores cuentan con herramientas que permiten a un administrador construir, implementar y configurar máquinas virtuales personalizadas basadas en una variedad de distribuciones de Linux. Muchas de estas compañías también tienen sistemas que permiten el despliegue y las migraciones de máquinas virtuales creadas desde la organización de un cliente.

Al evaluar la implementación de un sistema Linux en un entorno IaaS, hay algunos elementos claves que un administrador debe tener en cuenta:

Instancias de computación

Muchos proveedores de la nube cobrarán tasas de uso basadas en “instancias de computación”, o cuánto tiempo de CPU utilizará su infraestructura basada en la nube. Una planificación cuidadosa de cuánto tiempo de procesamiento requerirán realmente las aplicaciones ayudará a mantener manejables los costos de una solución en la nube.

Las instancias de computación a menudo también se refieren a la cantidad de máquinas virtuales que se aprovisionan en un entorno de nube. Una vez más, la mayor cantidad de instancias de sistemas que se ejecutan a la vez también influirá en la cantidad de tiempo total de CPU que se le cobrará a una organización.

Bloque de almacenamiento

Los proveedores de la nube también tienen varios niveles de almacenamiento en bloque disponibles para que una organización los use. Algunas ofertas están destinadas simplemente a ser un almacenamiento de red basado en la web para archivos, y otras ofertas se relacionan con el almacenamiento externo para una máquina virtual aprovisionada en la nube para usar y alojar archivos.

El costo de tales ofertas variará según la cantidad de almacenamiento utilizado y la velocidad del almacenamiento dentro de los centros de datos del proveedor. El acceso al almacenamiento más rápido generalmente costará más y, por el contrario, los datos "en reposo" (como en el

almacenamiento de archivos) a menudo son muy económicos.

Redes

Uno de los componentes principales de trabajar con un proveedor de soluciones en la nube es cómo se configurará la red virtual. Muchos proveedores de IaaS tendrán alguna forma de utilidades basadas en la web que pueden utilizarse para el diseño e implementación de diferentes rutas de red, subredes y configuraciones de firewall. Algunos incluso proporcionarán soluciones de DNS para que se puedan asignar FQDN de acceso público (nombres de dominio completos) a sus sistemas orientados a Internet. Incluso hay soluciones "híbridas" disponibles que pueden conectarse de una infraestructura de red existente en las instalaciones de su empresa a una infraestructura basada en la nube a través de una VPN (*virtual private network*), uniendo las dos infraestructuras.

Acceso seguro a los invitados (*guest*) en la nube

El método más frecuente para acceder a un invitado virtual remoto en una plataforma en la nube es mediante el uso del software OpenSSH. Un sistema Linux que reside en la nube tendría el servidor OpenSSH ejecutándose, mientras que un administrador usaría un cliente OpenSSH con claves precompartidas para acceso remoto.

Un administrador ejecutaría el siguiente comando

```
$ ssh-keygen
```

y seguiría las instrucciones para crear un par de claves SSH públicas y privadas. La clave privada permanece en el sistema local del administrador (almacenado en `~/.ssh/`) y la clave pública se copia en el sistema remoto de la nube, exactamente el mismo método que se usaría al trabajar con máquinas en red en una LAN corporativa.

El administrador entonces ejecutaría el siguiente comando:

```
$ ssh-copy-id -i <public_key> user@cloud_server
```

Esto copiará la clave SSH pública del par de claves recién generadas en el servidor remoto de la nube. La clave pública se registrará en el archivo `~/.ssh/authorized_keys` del servidor de la nube y establecerá los permisos apropiados en el archivo.

> **NOTE** Si solo hay un archivo de clave pública en el directorio `~/.ssh/`, entonces se puede omitir el modificador `-i`, ya que el comando `ssh-copy-id` pasará por defecto al archivo de clave pública en el directorio (normalmente el archivo que termina con la extensión `.pub`).

Algunos proveedores de la nube generarán automáticamente un par de claves cuando se aprovisione un nuevo sistema Linux. El administrador deberá descargar la clave privada para el nuevo sistema desde el proveedor de la nube y almacenarla en su sistema local. Tenga en cuenta que los permisos para las claves SSH deben ser `0600` para una clave privada y `0644` para una clave pública.

Preconfigurar sistemas en la nube

Una herramienta útil que simplifica los despliegues de máquinas virtuales basadas en la nube es la utilidad `cloud-init`. Este comando, junto con los archivos de configuración asociados y la imagen de máquina virtual predefinida, es un método independiente del proveedor para implementar un invitado (*guest*) Linux en una gran cantidad de proveedores de IaaS. Utilizando archivos de texto plano YAML (*YAML Ain't Markup Language*), un administrador puede preconfigurar configuraciones de red, selecciones de paquetes de software, configuración de claves SSH, creaciones de cuentas de usuario, configuraciones regionales, junto con una miríada de otras opciones para construir rápidamente nuevas sistemas.

Durante el arranque inicial de un nuevo sistema, `cloud-init` leerá la configuración de los archivos de configuración de YAML y los aplicará. Este proceso solo necesita aplicarse a la configuración inicial de un sistema y facilita la implementación de una flota de nuevos sistemas en la plataforma de un proveedor de la nube.

La sintaxis del archivo YAML utilizada con `cloud-init` se llama *cloud-config*. Aquí hay un archivo de ejemplo `cloud-config`:

```
#cloud-config
timezone: Africa/Dar_es_Salaam
hostname: test-system

# Update the system when it first boots up
apt_update: true
apt_upgrade: true

# Install the Nginx web server
packages:
 - nginx
```

Tenga en cuenta que en la línea superior no hay espacio entre el símbolo hash (`#`) y el término `cloud-config`.

> **NOTE** `cloud-init` no es solo para máquinas virtuales. El conjunto de herramientas `cloud-init` también se puede usar para preconfigurar contenedores (como los contenedores LXD Linux) antes de la implementación.

Contenedores

La tecnología de contenedores es similar en algunos aspectos a una máquina virtual, donde se obtiene un entorno aislado para implementar fácilmente una aplicación. Mientras que con una máquina virtual se emula una computadora completa, un contenedor utiliza el software suficiente para ejecutar una aplicación. De esta manera, hay mucho menos gastos generales.

Los contenedores permiten una mayor flexibilidad sobre la de una máquina virtual. Un contenedor de aplicaciones se puede migrar de un host a otro, al igual que una máquina virtual se puede migrar de un hipervisor a otro. Sin embargo, a veces una máquina virtual necesitará apagarse antes de que pueda migrarse, mientras que con un contenedor la aplicación siempre se está ejecutando mientras se migra. Los contenedores también facilitan la implementación de nuevas versiones de aplicaciones en conjunto con una versión existente. A medida que los usuarios cierran sus sesiones con contenedores en ejecución, el software de orquestación de contenedores puede eliminar automáticamente estos contenedores del sistema y reemplazarlos con la nueva versión, lo que reduce el tiempo de inactividad.

> **NOTE** Existen numerosas tecnologías de contenedor disponibles para Linux, como *Docker*, *Kubernetes*, *LXD/LXC*, *systemd-nspawn*, *OpenShift* y más. La implementación exacta de un paquete de software contenedor está más allá del alcance del examen LPIC-1.

Los contenedores utilizan el mecanismo *control groups* (mejor conocido como *cgroups*) dentro del kernel de Linux. El cgroup es una forma de particionar los recursos del sistema, como la memoria, el tiempo del procesador y el ancho de banda del disco y la red para una aplicación individual. Un administrador puede usar cgroups directamente para establecer límites de recursos del sistema en una aplicación, o un grupo de aplicaciones que podrían existir dentro de un solo cgroup. En esencia, esto es lo que hace el software contenedor para el administrador, además de proporcionar herramientas que facilitan la administración y la implementación de cgroups.

NOTE

Actualmente, el conocimiento de cgroups no es necesario para aprobar el examen LPIC-1. El concepto de cgroup se menciona aquí para que el candidato tenga al menos algunos conocimientos previos de cómo se segrega una aplicación en aras de la utilización de los recursos del sistema.

Ejercicios Guiados

1. ¿Qué extensiones de CPU son necesarias en una plataforma de hardware basada en x86 que ejecutará invitados (*guest*) totalmente virtualizados?

2. ¿Una instalación de un servidor de misión crítica que requerirá el rendimiento más rápido probablemente usará qué tipo de virtualización?

3. Dos máquinas virtuales que se han clonado a partir de la misma plantilla y que utilizan D-Bus funcionan de manera irregular. Ambos tienen nombres de host y configuraciones de red separadas. ¿Qué comando se usaría para determinar si cada una de las máquinas virtuales tienen diferentes ID de máquina D-Bus?

Ejercicios Exploratorios

1. Ejecute el siguiente comando para ver si su sistema ya tiene extensiones de CPU habilitadas para ejecutar una máquina virtual (sus resultados pueden variar según su CPU):

   ```
   grep --color -E "vmx|svm" /proc/cpuinfo
   ```

   Dependiendo de la salida, puede tener resaltado `vmx` (para CPU habilitadas con Intel VT-x) o `svm` resaltado (para CPU habilitadas con AMD SVM). Si no obtiene resultados, consulte las instrucciones de su BIOS o firmware UEFI sobre cómo habilitar la virtualización para su procesador.

2. Si su procesador admite virtualizaciones, busque la documentación de su distribución para ejecutar un hipervisor KVM.

   - Instale los paquetes necesarios para ejecutar un hipervisor KVM.

   - Si está utilizando un entorno de escritorio gráfico, se recomienda instalar también la aplicación `virt-manager`, que es una interfaz gráfica que se puede utilizar en una instalación de KVM. Esto ayudará en la instalación y gestión de máquinas virtuales.

   - Descargue una imagen ISO de la distribución de Linux de su elección y, siguiendo la documentación de su distribución, cree una nueva máquina virtual utilizando esta ISO.

Resumen

En esta lección cubrimos los conceptos básicos de máquinas virtuales y contenedores, y cómo estas tecnologías se pueden usar con Linux.

Describimos brevemente los siguientes comandos:

`dbus-uuidgen`
: Se utiliza para verificar y ver la ID de DBus de un sistema.

`ssh-keygen`
: Se utiliza para generar un par de claves SSH públicas y privadas para usar cuando se accede a sistemas remotos basados en la nube.

`ssh-copy-id`
: Se utiliza para copiar la clave SSH pública de un sistema en un sistema remoto para facilitar la autenticación remota.

`cloud-init`
: Se utiliza para ayudar en la configuración e implementación de máquinas virtuales y contenedores en un entorno de nube.

Respuestas a los ejercicios guiados

1. ¿Qué extensiones de CPU son necesarias en una plataforma de hardware basada en x86 que ejecutará invitados (*guest*) totalmente virtualizados?

   VT-x para Intel CPUs o AMD-V para AMD CPUs

2. ¿Una instalación de un servidor de misión crítica que requerirá el rendimiento más rápido probablemente usará qué tipo de virtualización?

   Un sistema operativo que utiliza la paravirtualización, como Xen, ya que el sistema operativo invitado puede hacer un mejor uso de los recursos de hardware disponibles es mediante el uso de controladores de software diseñados para trabajar con el hipervisor.

3. Dos máquinas virtuales que se han clonado a partir de la misma plantilla y que utilizan D-Bus funcionan de manera irregular. Ambos tienen nombres de host y configuraciones de red separadas. ¿Qué comando se usaría para determinar si cada una de las máquinas virtuales tienen diferentes ID de máquina D-Bus?

   `dbus-uuidgen --get`

Respuestas a ejercicios exploratorios

1. Ejecute el siguiente comando para ver si su sistema ya tiene extensiones de CPU habilitadas para ejecutar una máquina virtual (sus resultados pueden variar según su CPU): `grep --color -E "vmx|svm" /proc/cpuinfo`. Dependiendo de la salida, puede tener "vmx" resaltado (para CPU habilitadas con Intel VT-x) o "svm" resaltado (para CPU habilitadas con AMD SVM). Si no obtiene resultados, consulte las instrucciones de su BIOS o firmware UEFI sobre cómo habilitar la virtualización para su procesador.

   Los resultados variarán según la CPU que tenga. Aquí hay un ejemplo de salida de una computadora con una CPU Intel con extensiones de virtualización habilitadas en el firmware UEFI:

   ```
   $ grep --color -E "vmx|svm" /proc/cpuinfo
   flags       : fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov pat
   pse36 clflush dts acpi mmx fxsr sse sse2 ss ht tm pbe syscall nx pdpe1gb rdtscp
   lm constant_tsc art arch_perfmon pebs bts rep_good nopl xtopology nonstop_tsc
   cpuid aperfmperf pni pclmulqdq dtes64 monitor ds_cpl vmx smx est tm2 ssse3 sdbg
   fma cx16 xtpr pdcm pcid sse4_1 sse4_2 x2apic movbe popcnt tsc_deadline_timer
   aes xsave avx f16c rdrand lahf_lm abm 3dnowprefetch cpuid_fault epb
   invpcid_single pti ssbd ibrs ibpb stibp tpr_shadow vnmi flexpriority ept vpid
   ept_ad fsgsbase tsc_adjust bmi1 hle avx2 smep bmi2 erms invpcid rtm mpx rdseed
   adx smap clflushopt intel_pt xsaveopt xsavec xgetbv1 xsaves dtherm ida arat pln
   pts hwp hwp_notify hwp_act_window hwp_epp md_clear flush_l1d
   ```

2. Si su procesador admite virtualizaciones, busque la documentación de su distribución para ejecutar un hipervisor KVM.

   - Instale los paquetes necesarios para ejecutar un hipervisor KVM.

     Esto variará dependiendo de su distribución, pero aquí hay algunos puntos de partida:

     Ubuntu — https://help.ubuntu.com/lts/serverguide/libvirt.html

     Fedora — https://docs.fedoraproject.org/en-US/quick-docs/getting-started-with-virtualization/

     Arch Linux — https://wiki.archlinux.org/index.php/KVM

   - Si está utilizando un entorno de escritorio gráfico, se recomienda instalar también la aplicación `virt-manager`, que es una interfaz gráfica que se puede utilizar en una instalación de KVM. Esto ayudará en la instalación y gestión de máquinas virtuales.

Nuevamente, esto variará según la distribución. Un ejemplo usando Ubuntu se ve así:

```
$ sudo apt install virt-manager
```

◦ Descargue una imagen ISO de la distribución de Linux de su elección y, siguiendo la documentación de su distribución, cree una nueva máquina virtual utilizando esta ISO.

Esta tarea es manejada fácilmente por el paquete `virt-manager`. Sin embargo, se puede crear una máquina virtual desde la línea de comandos utilizando el comando `virt-install`. Pruebe ambos métodos para comprender cómo se implementan las máquinas virtuales.



Tema 103: Comandos GNU y Unix



103.1 Trabajar desde la línea de comandos

Referencia al objetivo del LPI

LPIC-1 version 5.0, Exam 101, Objective 103.1

Importancia

4

Áreas de conocimiento clave

- Usar comandos de shell individuales y secuencias de comandos de una línea para realizar tareas básicas en la línea de comandos.
- Usar y modificar el entorno de shell, lo que incluye definir, referenciar y exportar variables de entorno.
- Usar y editar el historial de comandos.
- Invocar comandos dentro y fuera de la ruta definida.

Lista parcial de archivos, términos y utilidades

- `bash`
- `echo`
- `env`
- `export`
- `pwd`
- `set`
- `unset`
- `type`

- `which`
- `man`
- `uname`
- `history`
- `.bash_history`
- Uso de comillas



103.1 Lección 1

Certificación:	LPIC-1
Versión:	5.0
Tema:	103 Comandos GNU y Unix
Objectivo:	103.1 Trabajando en la línea de comandos
Lección:	1 de 2

Introducción

Los recién llegados al mundo de la administración de Linux y el shell Bash a menudo se sienten un poco perdidos sin la comodidad tranquilizadora de una interfaz GUI. Están acostumbrados a tener acceso con el botón derecho sobre las señales visuales y la información contextual que las utilidades del administrador de archivos gráficos ponen a disposición. Por lo tanto, es importante aprender rápidamente y dominar el conjunto relativamente pequeño de herramientas de línea de comandos a través del cual puede aprovechar instantáneamente todos los datos ofrecidos por su antigua GUI y más.

Obteniendo información del sistema

Mientras observa el rectángulo parpadeante de una línea de comandos, su primera pregunta probablemente será “¿Dónde estoy?” O, más precisamente, “¿Dónde estoy en el sistema de archivos de Linux en este momento y si, por ejemplo, he creado un nuevo archivo, ¿dónde reside?” Lo que busca aquí es su *directorio de trabajo actual*, y el comando `pwd` te dirá lo que quieres saber:

```
$ pwd
/home/frank
```

Suponiendo que Frank está actualmente conectado al sistema y ahora está en su directorio de inicio: `/home/frank/`. Si Frank crea un archivo vacío usando el comando `touch` sin especificar ninguna otra ubicación en el sistema de archivos, el archivo se creará dentro de `/home/frank/`. Al listar el contenido del directorio usando `ls` nos mostrará ese nuevo archivo:

```
$ touch newfile
$ ls
newfile
```

Además de su ubicación en el sistema de archivos, a menudo querrá información sobre el sistema Linux que está ejecutando. Esto puede incluir el número exacto de lanzamiento de su distribución o la versión del kernel de Linux que está cargada actualmente. La herramienta `uname` es lo que busca aquí. Y, en particular, `uname` usando la opción `-a` ("all").

```
$ uname -a
Linux base 4.18.0-18-generic #19~18.04.1-Ubuntu SMP Fri Apr 5 10:22:13 UTC 2019
x86_64 x86_64 x86_64 GNU/Linux
```

Aquí, `uname` muestra que la máquina de Frank tiene instalada la versión 4.18.0 del kernel de Linux y ejecuta Ubuntu 18.04 en una CPU de 64 bits (`x86_64`).

Obteniendo información de los comandos

A menudo encontrará documentación que habla sobre comandos de Linux con los que aún no está familiarizado. La línea de comandos en sí ofrece todo tipo de información útil sobre qué hacen los comandos y cómo usarlos de manera efectiva. Quizás la información más útil se encuentra dentro de los muchos archivos del sistema `man`.

Como regla general, los desarrolladores de Linux escriben archivos `man` y los distribuyen junto con las utilidades que crean. Los archivos `man` son documentos altamente estructurados cuyo contenido está dividido intuitivamente por encabezados de sección estándar. Si escribe `man` seguido del nombre de un comando, obtendrá información que incluye el nombre del comando, una breve sinopsis de uso, una descripción más detallada y algunos antecedentes históricos y de licencia importantes. Aquí hay un ejemplo:

```
$ man uname
UNAME(1)             User Commands            UNAME(1)
NAME
   uname - print system information
SYNOPSIS
   uname [OPTION]...
DESCRIPTION
   Print certain system information.  With no OPTION, same as -s.
   -a, --all
      print  all information, in the following order, except omit -p
      and -i if unknown:
   -s, --kernel-name
      print the kernel name
   -n, --nodename
      print the network node hostname
   -r, --kernel-release
      print the kernel release
   -v, --kernel-version
      print the kernel version
   -m, --machine
      print the machine hardware name
   -p, --processor
      print the processor type (non-portable)
   -i, --hardware-platform
      print the hardware platform (non-portable)
   -o, --operating-system
      print the operating system
   --help display this help and exit
   --version
      output version information and exit
AUTHOR
   Written by David MacKenzie.
REPORTING BUGS
   GNU coreutils online help: <http://www.gnu.org/software/coreutils/>
   Report uname translation bugs to
   <http://translationproject.org/team/>
COPYRIGHT
   Copyright©2017 Free Software Foundation, Inc. License  GPLv3+: GNU
   GPL version 3 or later <http://gnu.org/licenses/gpl.html>.
   This is free software: you are free to change and redistribute it.
   There is NO WARRANTY, to the extent permitted by law.
SEE ALSO
   arch(1), uname(2)
   Full documentation at: <http://www.gnu.org/software/coreutils/uname>
   or available locally via: info '(coreutils) uname invocation'
```

```
GNU coreutils 8.28       January 2018            UNAME(1)
```

`man` solo funciona cuando se le proporciona un nombre de comando exacto. Sin embargo, si no está seguro del nombre del comando que está buscando, puede usar el comando `apropos` para buscar a través de los nombres y descripciones de la página `man`. Suponiendo, por ejemplo, que no puede recordar que es `uname` la que le dará su versión actual del kernel de Linux, puede pasar la palabra `kernel` a `apropros`. Probablemente obtendrá muchas líneas de salida, incluyendo estas:

```
$ apropos kernel
systemd-udevd-kernel.socket (8) - Device event managing daemon
uname (2)            - get name and information about current kernel
urandom (4)          - kernel random number source devices
```

Si no necesita la documentación completa de un comando, puede obtener rápidamente datos básicos sobre un comando usando `type`. Este ejemplo usa `type` para consultar cuatro comandos separados a la vez. Los resultados nos muestran que `cp` (“copy”) es un programa que reside en `/bin/cp` y que `kill` (cambiar el estado de un proceso en ejecución) es un shell incorporado, lo que significa que es en realidad una parte del shell Bash:

```
$ type uname cp kill which
uname is hashed (/bin/uname)
cp is /bin/cp
kill is a shell builtin
which is /usr/bin/which
```

Tenga en cuenta que, además de ser un comando binario regular como `cp`, `uname` también es “hash”. Esto se debe a que Frank recientemente utilizó "uname" y, para aumentar la eficiencia del sistema, se agregó a una tabla hash para hacer más accesible la próxima vez que se ejecute. Si ejecutara `type uname` después de un arranque del sistema, Frank encontraría que `type` una vez más describe `uname` como un binario regular.

NOTE Una forma más rápida de limpiar la tabla hash es ejecutar el comando `hash -d`.

A veces, especialmente cuando se trabaja con scripts automatizados, necesitará una fuente de información más simple sobre un comando. El comando `which`, el cual con `type` visualizamos anteriormente, no devolverá nada más que la ubicación absoluta de un comando. Este ejemplo localiza los comandos `uname` y `which`.

```
$ which uname which
/bin/uname
/usr/bin/which
```

NOTE Si desea mostrar información sobre los comandos “builtin”, puede usar el comando `help`.

Usando su historial de comandos

A menudo investigará cuidadosamente el uso adecuado de un comando y lo ejecutará con éxito junto con un complicado rastro de opciones y argumentos. Pero, ¿qué sucede unas semanas más tarde cuando necesita ejecutar el mismo comando con las mismas opciones y argumentos pero no puede recordarlo? En lugar de tener que comenzar su investigación nuevamente desde cero, a menudo podrá recuperar el comando original usando `history`.

Al escribir `history`, aparecerán los comandos más recientes que haya ejecutado, los cuáles aparecerán en último lugar. Puede buscar fácilmente a través de esos comandos canalizando una cadena específica al comando `grep`. Este ejemplo buscará cualquier comando que incluya el texto `bash_history`:

```
$ history | grep bash_history
1605 sudo find /home -name ".bash_history" | xargs grep sudo
```

Aquí se devuelve un solo comando junto con su número de secuencia, 1605.

Y hablando de `bash_history`, ese es en realidad el nombre de un archivo oculto que debe encontrar dentro del directorio de inicio de su usuario. Dado que es un archivo oculto (designado como tal por el punto que precede a su nombre de archivo), solo será visible al enumerar el contenido del directorio usando `ls` con el argumento `-a`:

```
$ ls /home/frank
newfile
$ ls -a /home/frank
.  ..  .bash_history  .bash_logout  .bashrc  .profile  .ssh  newfile
```

¿Qué hay en el archivo `.bash_history`? Eche un vistazo por sí mismo: verá cientos y cientos de sus comandos más recientes. Sin embargo, es posible que se sorprenda al descubrir que faltan algunos de sus comandos más recientes. Esto se debe a que, aunque se agregan instantáneamente a la base de

datos dinámica `history`, las últimas adiciones a su historial de comandos no se escriben en el archivo `.bash_history` hasta que salga de la sesión.

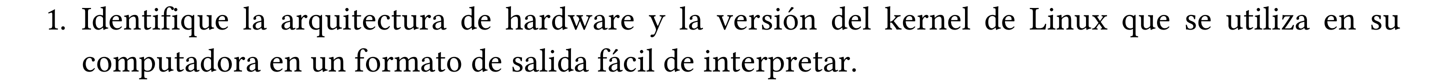
Puede aprovechar el contenido de `history` para hacer que su experiencia en la línea de comandos sea mucho más rápida y eficiente usando las teclas de flecha arriba y abajo en su teclado. Al presionar la tecla hacia arriba varias veces se completará la línea de comandos con comandos recientes. Cuando llegue al que desea ejecutar por segunda vez, puede ejecutarlo presionando *Enter*. Esto facilita la recuperación y, si lo desea, la modificación de comandos varias veces durante una sesión de shell.

Ejercicios Guiados

1. Utilice el sistema `man` para determinar cómo decirle a `apropos` que envíe un comando breve para que solo envíe un mensaje corto de uso y luego salga.

2. Utilice el sistema `man` para determinar qué licencia de copyright se asigna al comando `grep`.

Ejercicios Exploratorios



1. Identifique la arquitectura de hardware y la versión del kernel de Linux que se utiliza en su computadora en un formato de salida fácil de interpretar.

2. Imprima las últimas veinte líneas de la base de datos dinámica `history` y el archivo `.bash_history` para compararlos.

3. Utilice la herramienta `apropos` para identificar la página `man` donde encontrará el comando que necesitará para mostrar el tamaño de un dispositivo de bloque físico conectado en bytes en lugar de megabytes o gigabytes.

Resumen

En esta lección aprendimos:

- Cómo obtener información sobre la ubicación de su sistema de archivos y la pila de software del sistema operativo.
- Cómo encontrar ayuda para el uso de comandos.
- Cómo identificar la ubicación del sistema de archivos y el tipo de comandos binarios.
- Cómo encontrar y reutilizar comandos ejecutados previamente.

Los siguientes comandos se discutieron en esta lección:

`pwd`
: Imprime la ruta al directorio de trabajo actual.

`uname`
: Imprime la arquitectura de hardware de su sistema, la versión del kernel de Linux, la distribución y la versión de distribución.

`man`
: Accede a los archivos de ayuda que documentan el uso de comandos.

`type`
: Imprimer la ubicación del sistema de archivos y el tipo de uno o más comandos.

`which`
: Imprime la ubicación del sistema de archivos para un comando.

`history`
: Imprime o reutiliza comandos que hayan ejecutado previamente.

Respuestas a los ejercicios guiados

1. Utilice el sistema `man` para determinar cómo decirle a `apropos` que envíe un comando breve para que solo envíe un mensaje corto de uso y luego salga.

   Ejecute `man apropos` y desplácese hacia abajo por la sección “Options” hasta llegar al párrafo `--usage`.

2. Utilice el sistema `man` para determinar qué licencia de copyright se asigna al comando `grep`.

   Ejecute `man grep` y desplácese hacia abajo a la sección “Copyright” del documento. Tenga en cuenta que el programa utiliza los derechos de autor de la Free Software Foundation.

Respuestas a ejercicios exploratorios

1. Identifique la arquitectura de hardware y la versión del kernel de Linux que se utiliza en su computadora en un formato de salida fácil de Interpretar.

   Ejecute `man uname`, lea la sección “Description” e identifique los argumentos del comando que se mostrarán solo los resultados exactos que desea. Observe cómo `-r` le dará la versión del kernel y `-i` le proporcionará una plataforma de hardware.

   ```
   $ man uname
   $ uname -v
   $ uname -i
   ```

2. Imprima las últimas veinte líneas de la base de datos dinámica `history` y el archivo `.bash_history` para compararlos.

   ```
   $ history 20
   $ tail -n 20 .bash_history
   ```

3. Utilice la herramienta `apropos` para identificar la página `man` donde encontrará el comando que necesitará para mostrar el tamaño de un dispositivo de bloque físico conectado en bytes en lugar de megabytes o gigabytes.

   Una forma sería ejecutar `apropos` con la cadena `block`, leer los resultados, observar que `lsblk` enumera los dispositivos de bloque (y, por lo tanto, sería la herramienta más adecuada para nuestras necesidades), ejecutar `man lsblk`, desplácese por la sección “Description” y observe que `-b` mostrará el tamaño del dispositivo en bytes. Finalmente, ejecute `lsblk -b` para visualizar la salida.

   ```
   $ apropos block
   $ man lsblk
   $ lsblk -b
   ```



103.1 Lección 2

Certificación:	LPIC-1
Versión:	5.0
Tema:	103 Comandos GNU y Unix
Objectivo:	103.1 Trabajando en la línea de comandos
Lección:	2 de 2

Introducción

Un entorno de sistema operativo incluye las herramientas básicas, como shells de línea de comandos y, a veces, una GUI, que necesitará para hacer las cosas. Pero su entorno también vendrá con un catálogo de accesos directos y valores preestablecidos. Aquí es donde aprenderemos cómo listar, invocar y administrar esos valores.

Encontrar las variables de entorno

Entonces, ¿cómo identificamos los valores actuales para cada una de nuestras variables de entorno? Una forma es a través del comando `env`:

```
$ env
DBUS_SESSION_BUS_ADDRESS=unix:path=/run/user/1000/bus
XDG_RUNTIME_DIR=/run/user/1000
XAUTHORITY=/run/user/1000/gdm/Xauthority
XDG_CONFIG_DIRS=/etc/xdg/xdg-ubuntu:/etc/xdg
PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/l
ocal/games:/snap/bin
GJS_DEBUG_TOPICS=JS ERROR;JS LOG
[...]
```

Obtendrá una gran cantidad de resultados, mucho más de lo que se incluye en el extracto anterior. Pero por ahora tenga en cuenta la entrada `PATH`, que contiene los directorios donde su shell (y otros programas) buscarán otros programas sin tener que especificar una ruta completa. Con ese conjunto, podría ejecutar un programa binario que reside, por ejemplo, en `/usr/local/bin` desde su directorio de inicio y se ejecutaría como si el archivo fuera local.

Cambiemos de tema por un momento. El comando `echo` imprimirá en la pantalla lo que usted le indique. Lo crea o no, habrá muchas veces que será muy útil que `echo` repita literalmente algo.

```
$ echo "Hi. How are you?"
Hi. How are you?
```

Pero hay algo más que puede hacer con `echo`. Cuando le da el nombre de una variable de entorno, y le dice que esta es una variable al prefijar el nombre de la variable con un `$`, entonces, en lugar de simplemente imprimir el nombre de la variable, el shell lo expandirá dándole el valor. ¿No está seguro de si su directorio favorito está actualmente en la ruta? Puede verificar rápidamente ejecutándolo a través de `echo`:

```
$ echo $PATH
/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/local/
games:/snap/bin
```

Crear nuevas variables de entorno

Puede agregar sus propias variables personalizadas a su entorno. La forma más simple es usar el caracter `=`. La cadena a la izquierda será el nombre de su nueva variable, y la cadena a la derecha será su valor. Ahora puede verificar el nombre de la variable con `echo` para confirmar que funcionó:

```
$ myvar=hello
$ echo $myvar
hello
```

NOTE Observe que no hay espacio a ambos lados del signo igual durante la asignación de variables.

¿Pero realmente funcionó? Escriba `bash` en la terminal para abrir un nuevo shell. Este nuevo shell se ve exactamente como el que acababa de encontrar, pero en realidad es un hijo (*child*) del original (al que llamamos *parent*). Ahora, dentro de esta nueva consola hija, intente obtener el valor con `echo`. Nada. ¿Qué está pasando?

```
$ bash
$ echo $myvar

$
```

Una variable creada de la manera que acabamos de hacerlo solo estará disponible localmente, dentro de la sesión de shell inmediata. Si inicia un nuevo shell, o cierra la sesión con `exit`, la variable no lo acompañará. Si escribe `exit` aquí, volverá a su shell principal original que, en este momento, es donde queremos estar. Puede ejecutar `echo $myvar` una vez más si lo desea solo para confirmar que la variable sigue siendo válida. Ahora escriba `export myvar` para pasar la variable a los shells secundarios que pueda abrir posteriormente. Pruébelo: teclee `bash` para un nuevo shell y luego `echo`:

```
$ exit
$ export myvar
$ bash
$ echo $myvar
hello
```

Todo esto puede parecer un poco útil cuando estamos creando shells sin ningún propósito real. Pero comprender cómo se propagan las variables de shell a través de su sistema será muy importante una vez que comience a escribir scripts.

Eliminar variables de entorno

¿Quiere saber cómo limpiar todas esas variables que ha creado? Una forma es simplemente cerrar su shell principal o reiniciar su computadora. Pero hay formas más simples. Como, por ejemplo, `unset`. Al escribir `unset` (sin el `$`) se eliminará la variable. Con `echo` podemos verificarlo.

```
$ unset myvar
$ echo $myvar

$
```

Si existe un comando `unset`, entonces existe un comando `set`. Ejecutar `set` por sí solo mostrará una gran cantidad de resultados, pero en realidad no es tan diferente de lo que `env` muestra. Mire la primera línea de salida que obtendrá cuando filtre por `PATH`:

```
$ set | grep PATH
PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/l
ocal/games:/snap/bin
[...]
```

¿Cuál es la diferencia entre `set` y `env`? Para nuestros propósitos, lo principal es que `set` generará todas las variables y funciones. Vamos a ilustrar eso. Crearemos una nueva variable llamada `mynewvar` y luego confirmaremos que está allí:

```
$ mynewvar=goodbye
$ echo $mynewvar
goodbye
```

Ahora, ejecutar `env` mientras se usa `grep` para filtrar la cadena `mynewvar` no mostrará ningún resultado. Pero ejecutar `set` de la misma manera nos mostrará nuestra variable local.

```
$ env | grep mynewvar

$ set | grep mynewvar
mynewvar=goodbye
```

Mantener el Valor de Caracteres Especiales

Ahora es un buen momento para presentar el problema de los caracteres especiales. Bash normalmente leerá literalmente los caracteres alfanuméricos (a-z y 0-9). Si intenta crear un nuevo archivo llamado `myfile`, simplemente escriba `touch` seguido de `myfile` y Bash sabrá qué hacer con él. Pero si desea incluir un carácter especial en su nombre de archivo, deberá hacerlo de una forma diferente.

Para ilustrar esto, escribiremos `touch` y lo definimos el nombre: `my big file`. El problema es que hay dos espacios entre las palabras que Bash interpretará. Si bien, técnicamente, no llamarías a un espacio un “caracter”, uno esperaría que Bash lo entendiera de esa forma. Si lista el contenido de su directorio actual, en lugar de un archivo llamado `my big file`, verá tres archivos denominados, respectivamente, `my`, `big` y `file`. Esto se debe a que Bash interpretó que deseaba crear varios archivos cuyos nombres estaba pasando en una lista:

```
$ touch my big file
$ ls
my big file
```

Los espacios se interpretarán de la misma manera si elimina (`rm`) los tres archivos, todo en un comando:

```
$ rm my big file
```

Ahora probémoslo de la manera correcta. Escriba `touch` y las tres partes de su nombre de archivo, pero esta vez incluya el nombre entre comillas dobles. Esta vez funcionó. Listar el contenido del directorio le mostrará un solo archivo con el nombre apropiado.

```
$ touch "my big file"
$ ls
'my big file'
```

Hay otras formas de obtener el mismo efecto. Las comillas simples, por ejemplo, funcionan tan bien como las comillas dobles. (Tenga en cuenta que las comillas simples conservarán el valor literal de todos los caracteres, mientras que las comillas dobles conservarán todos los caracteres *excepto* para `$`, `` ` ``, `\` y, en ciertos casos, `!`.)

```
$ rm 'my big file'
```

Anteponer cada carácter especial con la barra invertida “escape” hará que Bash lo lea literalmente.

```
$ touch my\ big\ file
```


Ejercicios Guiados

1. Use el comando `export` para agregar un nuevo directorio a su ruta (este no debe sobrevivir a un reinicio).

2. Use el comando `unset` para eliminar la variable `PATH`. Intente ejecutar un comando (como `sudo cat /etc/shadow`) usando `sudo`. ¿Que pasó? ¿Por qué? (Al salir de su shell, volverá a su estado original).

Ejercicios Exploratorios

1. Busque en Internet para encontrar y explorar la lista completa de caracteres especiales.
2. Intente ejecutar comandos usando cadenas formadas por caracteres especiales y usando varios métodos para escaparlos. ¿Hay diferencias entre la forma en que se comportan esos métodos?

Resumen

En esta lección aprendimos:

- ¿Cómo identificar las variables de entorno de su sistema?
- ¿Cómo crear sus propias variables de entorno y exportarlas a otros shells?
- ¿Cómo eliminar variables de entorno y cómo usar los comandos `env` y `set`?
- ¿Cómo escapar caracteres especiales para que Bash los lea literalmente?

Los siguientes comandos se discutieron en esta lección:

`echo`
: Imprime cadenas de entrada y variables.

`env`
: Entiende y modifica sus variables de entorno.

`export`
: Pasa una variable de entorno a los shells secundarios.

`unset`
: Borra valores y atributos de variables y funciones de shell.

Respuestas a los ejercicios guiados

1. Use el comando `export` para agregar un nuevo directorio a su ruta (esto no debe sobrevivir a un reinicio).

   Puede agregar temporalmente un nuevo directorio (quizás uno llamado `myfiles` que se encuentra en su directorio de inicio) a su ruta usando `export PATH="/home/yourname/myfiles:$PATH"`. Cree un script simple en el directorio `myfiles/`, hágalo ejecutable e intente ejecutarlo desde un directorio diferente. Estos comandos suponen que está en su directorio de inicio que contiene un directorio llamado `myfiles`.

   ```
   $ touch myfiles/myscript.sh
   $ echo '#!/bin/bash' >> myfiles/myscript.sh
   $ echo 'echo Hello' >> myfiles/myscript.sh
   $ chmod +x myfiles/myscript.sh
   $ myscript.sh
   Hello
   ```

2. Use el comando `unset` para eliminar la variable `PATH`. Intente ejecutar un comando (como `sudo cat /etc/shadow`) usando `sudo`. ¿Que pasó? ¿Por qué? (Al salir de su shell, volverá a su estado original).

   Al escribir `unset PATH` se borrará la configuración de ruta actual. Intentar invocar un binario sin su dirección absoluta fallará. Por esa razón, intentar ejecutar un comando usando sudo (que en sí mismo es un programa binario ubicado en `/usr/bin/sudo`) fallará, a menos que especifique la ruta absoluta, como en: `/usr/bin/sudo/bin/cat/etc/shadow`. Puede restablecer su `PATH` usando `export` o simplemente salga del shell.

Respuestas a ejercicios exploratorios

1. Busque en Internet para encontrar y explorar la lista completa de caracteres especiales.

   Lista de ejemplo: `& ; | * ? " ' [ ] ( ) $ < > { } # / \ ! ~`.

2. Intente ejecutar comandos usando cadenas formadas por caracteres especiales y usando varios métodos para escaparlos. ¿Hay diferencias entre la forma en que se comportan esos métodos?

   Escapar usando caracteres `"` conservará los valores especiales del signo de dólar, una barra invertida y la barra invertida. Escapar usando un carácter `'` hará que todos los caracteres sean literales.

   ```
   $ echo "$mynewvar"
   goodbye
   $ echo '$mynewvar'
   $mynewvar
   ```



103.2 Procesar secuencias de texto usando filtros

Referencia al objetivo del LPI

LPIC-1 v5, Exam 101, Objective 103.2

Importancia

2

Áreas de conocimiento clave

- Enviar archivos de texto y flujos de salida a través de filtros de utilidades de texto para modificar la salida usando comandos UNIX estándar incluidos en el paquete GNU textutils.

Lista parcial de archivos, términos y utilidades

- `bzcat`
- `cat`
- `cut`
- `head`
- `less`
- `md5sum`
- `nl`
- `od`
- `paste`
- `sed`
- `sha256sum`
- `sha512sum`

- `sort`
- `split`
- `tail`
- `tr`
- `uniq`
- `wc`
- `xzcat`
- `zcat`



103.2 Lección 1

Certificación:	LPIC-1
Versión:	5.0
Tema:	103 Comandos GNU y Unix
Objetivo:	103.2 Procesar secuencias de texto usando filtros
Lección:	1 de 1

Introducción

Tratar con el texto es una parte importante del trabajo de cada administrador de sistemas. Doug McIlroy, miembro del equipo de desarrollo original de Unix, resumió la filosofía de Unix y dijo (entre otras cosas importantes): "Escribir programas para manejar flujos de texto, porque esa es una interfaz universal". Linux está inspirado en el funcionamiento de Unix y adopta firmemente su filosofía, por lo que un administrador debe esperar muchas herramientas de manipulación de texto dentro de una distribución de Linux.

Una revisión rápida sobre redirecciones y tuberías (Pipes)

También de la filosofía Unix:

- Escriba programas que hagan una cosa y que la hagan bien.
- Escribir programas para trabajar juntos.

Una forma importante de hacer que los programas funcionen juntos es a través de *piping* y *redirections*. Casi todos sus programas de manipulación de texto obtendrán texto de una entrada estándar (*stdin*), lo enviarán a una salida estándar (*stdout*) y enviarán eventuales errores a una salida

de error estándar (*stderr*). A menos que especifique lo contrario, la entrada estándar será la que escriba en su teclado (el programa lo leerá después de presionar la tecla Enter). Del mismo modo, la salida estándar y los errores se mostrarán en la pantalla de su terminal. Veamos cómo funciona esto.

En su terminal, teclee `cat` y luego presione la tecla Enter. Luego teclee un texto al azar.

```
$ cat
This is a test
This is a test
Hey!
Hey!
It is repeating everything I type!
It is repeating everything I type!
(I will hit ctrl+c so I will stop this nonsense)
(I will hit ctrl+c so I will stop this nonsense)
^C
```

Para obtener más información sobre el comando `cat` (el término proviene de “concatenate”), consulte las páginas del manual.

NOTE Si está trabajando en una instalación realmente simple de un servidor Linux, algunos comandos como `info` y `less` podrían no estar disponibles. Si este es el caso, instale estas herramientas utilizando el procedimiento adecuado en su sistema como se describe en las lecciones correspondientes.

Como se demostró anteriormente, si no especifica de dónde debe leer `cat`, leerá desde la entrada estándar (lo que escriba) y enviará lo que lea a su ventana de terminal (su salida estándar).

Ahora intente lo siguiente:

```
$ cat > mytextfile
This is a test
I hope cat is storing this to mytextfile as I redirected the output
I will hit ctrl+c now and check this
^C

$ cat mytextfile
This is a test
I hope cat is storing this to mytextfile as I redirected the output
I will hit ctrl+c now and check this
```

El `>` (mayor que) indica a `cat` que dirija su salida al archivo `mytextfile`, no a la salida estándar. Ahora intente esto:

```
$ cat mytextfile > mynewtextfile
$ cat mynewtextfile
This is a test
I hope cat is storing this to mytextfile as I redirected the output
I will hit ctrl+c now and check this
```

Esto tiene el efecto de copiar `mytextfile` a `mynewtextfile`. En realidad, puede verificar que estos dos archivos tengan el mismo contenido realizando un `diff`:

```
$ diff mynewtextfile mytextfile
```

Como no hay salida, los archivos son iguales. Ahora intente con el operador de redireccionamiento anexado (`>>`):

```
$ echo 'This is my new line' >> mynewtextfile
$ diff mynewtextfile mytextfile
4d3
< This is my new line
```

Hasta ahora hemos usado redirecciones para crear y manipular archivos. También podemos usar tuberías (representadas por el símbolo `|`) para redirigir la salida de un programa a otro. Encontremos las líneas donde se encuentra la palabra "this":

```
$ cat mytextfile | grep this
I hope cat is storing this to mytextfile as I redirected the output
I will hit ctrl+c now and check this

$ cat mytextfile | grep -i this
This is a test
I hope cat is storing this to mytextfile as I redirected the output
I will hit ctrl+c now and check this
```

Ahora hemos canalizado la salida de `cat` a otro comando: `grep`. Note que cuando ignoramos mayúsculas y minúsculas (usando la opción `-i`) obtenemos una línea extra como resultado.

Procesando flujos de texto

Leer un archivo comprimido

Crearemos un archivo llamado `ftu.txt` que contenga una lista de los siguientes comandos:

```
bzcat
cat
cut
head
less
md5sum
nl
od
paste
sed
sha256sum
sha512sum
sort
split
tail
tr
uniq
wc
xzcat
zcat
```

Ahora usaremos el comando `grep` para imprimir todas las líneas que contienen la cadena `cat`:

```
$ cat ftu.txt | grep cat
bzcat
cat
xzcat
zcat
```

Otra forma de obtener esta información es simplemente usar el comando `grep` para filtrar el texto directamente, sin la necesidad de usar otra aplicación para enviar el flujo de texto a `stdout`.

```
$ grep cat ftu.txt
bzcat
cat
xzcat
zcat
```

NOTE Recuerde que hay muchas formas de realizar la misma tarea con Linux.

Hay otros comandos que manejan archivos comprimidos (`bzcat` para archivos comprimidos `bzip`, `xzcat` para archivos comprimidos `xz` y `zcat` para archivos comprimidos `gzip`) y cada uno se usa para ver el contenido de un archivo comprimido basado en el algoritmo de compresión utilizado.

Verifique que el archivo recién creado `ftu.txt` sea el único en el directorio, luego cree una versión comprimida `gzip` del archivo:

```
$ ls ftu*
ftu.txt

$ gzip ftu.txt
$ ls ftu*
ftu.txt.gz
```

Luego, use el comando `zcat` para ver el contenido del archivo comprimido con gzip:


```
$ zcat ftu.txt.gz
bzcat
cat
cut
head
less
md5sum
nl
od
paste
sed
sha256sum
sha512sum
sort
split
tail
tr
uniq
wc
xzcat
zcat
```

Tenga en cuenta que `gzip` comprimirá `ftu.txt` en `ftu.txt.gz` y eliminará el archivo original. Por defecto, no se mostrará ningún resultado del comando `gzip`. Sin embargo, si desea que `gzip` le diga qué está haciendo, use la opción `-v` para la salida "verbose".

Ver un archivo paginado

Usted sabe que `cat` concatena un archivo a la salida estándar (una vez que se proporciona un archivo después del comando). El archivo `/var/log/syslog` es donde su sistema Linux almacena todo lo importante que sucede en su sistema. Usando el comando `sudo` para elevar los privilegios para poder leer el archivo `/var/log/syslog`:

```
$ sudo cat /var/log/syslog
```

…verá mensajes que se desplazan muy rápido dentro de la ventana de su terminal. Puede canalizar la salida al programa `less` para que los resultados se paginen. Al usar `less` puede usar las teclas de flecha para navegar a través de la salida y también usar comandos similares a `vi` para navegar y buscar en todo el texto.

Sin embargo, en lugar de canalizar el comando `cat` en un programa de paginación, es más

pragmático usar el programa de paginación directamente:

```
$ sudo less /var/log/syslog
... (salida omitida para mayor claridad)
```

Obtener una parte de un archivo de texto

Si solo es necesario revisar el inicio o el final de un archivo, hay otros métodos disponibles. El comando `head` se usa para leer las primeras diez líneas de un archivo de manera predeterminada, y el comando `tail` se usa para leer las últimas diez líneas de un archivo de manera predeterminada. Ahora intente:

```
$ sudo head /var/log/syslog
Nov 12 08:04:30 hypatia rsyslogd: [origin software="rsyslogd" swVersion="8.1910.0"
x-pid="811" x-info="https://www.rsyslog.com"] rsyslogd was HUPed
Nov 12 08:04:30 hypatia systemd[1]: logrotate.service: Succeeded.
Nov 12 08:04:30 hypatia systemd[1]: Started Rotate log files.
Nov 12 08:04:30 hypatia vdr: [928] video directory scanner thread started (pid=882,
tid=928, prio=low)
Nov 12 08:04:30 hypatia vdr: [882] registered source parameters for 'A - ATSC'
Nov 12 08:04:30 hypatia vdr: [882] registered source parameters for 'C - DVB-C'
Nov 12 08:04:30 hypatia vdr: [882] registered source parameters for 'S - DVB-S'
Nov 12 08:04:30 hypatia vdr: [882] registered source parameters for 'T - DVB-T'
Nov 12 08:04:30 hypatia vdr[882]: vdr: no primary device found - using first
device!
Nov 12 08:04:30 hypatia vdr: [929] epg data reader thread started (pid=882,
tid=929, prio=high)
$ sudo tail /var/log/syslog
Nov 13 10:24:45 hypatia kernel: [ 8001.679238] mce: CPU7: Core temperature/speed
normal
Nov 13 10:24:46 hypatia dbus-daemon[2023]: [session uid=1000 pid=2023] Activating
via systemd: service name='org.freedesktop.Tracker1.Miner.Extract' unit='tracker-
extract.service' requested by ':1.73' (uid=1000 pid=2425
comm="/usr/lib/tracker/tracker-miner-fs ")
Nov 13 10:24:46 hypatia systemd[2004]: Starting Tracker metadata extractor...
Nov 13 10:24:47 hypatia dbus-daemon[2023]: [session uid=1000 pid=2023] Successfully
activated service 'org.freedesktop.Tracker1.Miner.Extract'
Nov 13 10:24:47 hypatia systemd[2004]: Started Tracker metadata extractor.
Nov 13 10:24:54 hypatia kernel: [ 8010.462227] mce: CPU0: Core temperature above
threshold, cpu clock throttled (total events = 502907)
Nov 13 10:24:54 hypatia kernel: [ 8010.462228] mce: CPU4: Core temperature above
threshold, cpu clock throttled (total events = 502911)
Nov 13 10:24:54 hypatia kernel: [ 8010.469221] mce: CPU0: Core temperature/speed
normal
Nov 13 10:24:54 hypatia kernel: [ 8010.469222] mce: CPU4: Core temperature/speed
normal
Nov 13 10:25:03 hypatia systemd[2004]: tracker-extract.service: Succeeded.
```

Para ayudar a ilustrar el número de líneas que se muestran, podemos canalizar la salida del comando `head` al comando `nl`, que mostrará el número de líneas de texto transmitidas al comando:

```
$ sudo head /var/log/syslog | nl
1   Nov 12 08:04:30 hypatia rsyslogd: [origin software="rsyslogd"
swVersion="8.1910.0" x-pid="811" x-info="https://www.rsyslog.com"] rsyslogd was
HUPed
2   Nov 12 08:04:30 hypatia systemd[1]: logrotate.service: Succeeded.
3   Nov 12 08:04:30 hypatia systemd[1]: Started Rotate log files.
4   Nov 12 08:04:30 hypatia vdr: [928] video directory scanner thread started
(pid=882, tid=928, prio=low)
5   Nov 12 08:04:30 hypatia vdr: [882] registered source parameters for 'A - ATSC'
6   Nov 12 08:04:30 hypatia vdr: [882] registered source parameters for 'C - DVB-C'
7   Nov 12 08:04:30 hypatia vdr: [882] registered source parameters for 'S - DVB-S'
8   Nov 12 08:04:30 hypatia vdr: [882] registered source parameters for 'T - DVB-T'
9   Nov 12 08:04:30 hypatia vdr[882]: vdr: no primary device found - using first
device!
10  Nov 12 08:04:30 hypatia vdr: [929] epg data reader thread started (pid=882,
tid=929, prio=high)
```

Y podemos hacer lo mismo canalizando la salida del comando `tail` al comando `wc`, que por defecto contará el número de palabras dentro de un documento, y usando la opción `-l` para imprimir el número de líneas de texto que el comando ha leído:

```
$ sudo tail /var/log/syslog | wc -l
10
```

Si un administrador necesita revisar más (o menos) del comienzo o el final de un archivo, la opción `-n` puede usarse para limitar la salida del comando:

```
$ sudo tail -n 5 /var/log/syslog
Nov 13 10:37:24 hypatia systemd[2004]: tracker-extract.service: Succeeded.
Nov 13 10:37:42 hypatia dbus-daemon[2023]: [session uid=1000 pid=2023] Activating
via systemd: service name='org.freedesktop.Tracker1.Miner.Extract' unit='tracker-
extract.service' requested by ':1.73' (uid=1000 pid=2425
comm="/usr/lib/tracker/tracker-miner-fs ")
Nov 13 10:37:42 hypatia systemd[2004]: Starting Tracker metadata extractor...
Nov 13 10:37:43 hypatia dbus-daemon[2023]: [session uid=1000 pid=2023] Successfully
activated service 'org.freedesktop.Tracker1.Miner.Extract'
Nov 13 10:37:43 hypatia systemd[2004]: Started Tracker metadata extractor.
$ sudo head -n 12 /var/log/syslog
Nov 12 08:04:30 hypatia rsyslogd: [origin software="rsyslogd" swVersion="8.1910.0"
x-pid="811" x-info="https://www.rsyslog.com"] rsyslogd was HUPed
Nov 12 08:04:30 hypatia systemd[1]: logrotate.service: Succeeded.
Nov 12 08:04:30 hypatia systemd[1]: Started Rotate log files.
Nov 12 08:04:30 hypatia vdr: [928] video directory scanner thread started (pid=882,
tid=928, prio=low)
Nov 12 08:04:30 hypatia vdr: [882] registered source parameters for 'A - ATSC'
Nov 12 08:04:30 hypatia vdr: [882] registered source parameters for 'C - DVB-C'
Nov 12 08:04:30 hypatia vdr: [882] registered source parameters for 'S - DVB-S'
Nov 12 08:04:30 hypatia vdr: [882] registered source parameters for 'T - DVB-T'
Nov 12 08:04:30 hypatia vdr[882]: vdr: no primary device found - using first
device!
Nov 12 08:04:30 hypatia vdr: [929] epg data reader thread started (pid=882,
tid=929, prio=high)
Nov 12 08:04:30 hypatia vdr: [882] no DVB device found
Nov 12 08:04:30 hypatia vdr: [882] initializing plugin: vnsiserver (1.8.0): VDR-
Network-Streaming-Interface (VNSI) Server
```

Los fundamentos de sed, el Editor de Stream

Echemos un vistazo a los otros archivos, términos y utilidades que no tienen `cat` en sus nombres. Podemos hacer esto pasando la opción `-v` a `grep`, que indica al comando que solo muestre las líneas que no contienen `cat`:

```
$ zcat ftu.txt.gz | grep -v cat
cut
head
less
md5sum
nl
od
paste
sed
sha256sum
sha512sum
sort
split
tail
tr
uniq
wc
```

La mayor parte de lo que podemos hacer con `grep` también podemos hacerlo con `sed`, el editor de flujo para filtrar y transformar texto (como se indica en la página del manual `sed`). Primero recuperaremos nuestro archivo `ftu.txt` descomprimiendo nuestro archivo `gzip` del archivo:

```
$ gunzip ftu.txt.gz
$ ls ftu*
ftu.txt
```

Ahora, podemos usar `sed` para listar solo las líneas que contienen la cadena `cat`:

```
$ sed -n /cat/p < ftu.txt
bzcat
cat
xzcat
zcat
```

Hemos utilizado el signo menor que `<` para dirigir el contenido del archivo `ftu.txt a` al comando `sed`. La palabra encerrada entre barras (es decir, `/cat/`) es el término que estamos buscando. La opción `-n` instruye a `sed` para que no produzca salida (a menos por las instrucciones posteriores del comando `p`). Intente ejecutar este mismo comando sin la opción `-n` para ver qué sucede. Entonces intente esto:

```
$ sed /cat/d < ftu.txt
cut
head
less
md5sum
nl
od
paste
sed
sha256sum
sha512sum
sort
split
tail
tr
uniq
wc
```

Si no usamos la opción `-n`, `sed` imprimirá todo desde el archivo, excepto lo que la `d` indica a `sed` que elimine de su salida.

Un uso común de `sed` es buscar y reemplazar texto dentro de un archivo. Suponga que desea cambiar cada aparición de `cat` a `dog`. Puede usar `sed` para hacer esto proporcionando la opción `s` para intercambiar cada instancia del primer término, `cat`, por el segundo término, `dog`:

```
$ sed s/cat/dog/ < ftu.txt
bzdog
dog
cut
head
less
md5sum
nl
od
paste
sed
sha256sum
sha512sum
sort
split
tail
tr
uniq
wc
xzdog
zdog
```

En lugar de utilizar un operador de redireccionamiento (`<`) para pasar el archivo `ftu.txt` a nuestro comando `sed`, podemos hacer que el comando `sed` opere directamente en el archivo. Lo intentaremos a continuación, mientras creamos simultáneamente una copia de seguridad del archivo original:

```
$ sed -i.backup s/cat/dog/ ftu.txt
$ ls ftu*
ftu.txt  ftu.txt.backup
```

La opción `-i` realizará una operación `sed` directamente en el archivo *original*. Si no utiliza el `.backup` después del parámetro `-i`, simplemente sobreescribirá su archivo *original*. Cualquier cosa que use como texto después del parámetro `-i` será el nombre con el que se guardará el archivo original antes de las modificaciones que le solicitó a `sed`.

Garantizar la integridad de los datos

Hemos demostrado lo fácil que es manipular archivos en Linux. Hay momentos en los que es posible que desee compartir un archivo con otra persona, y desea asegurarse de que el destinatario termine con una copia verdadera del archivo original. Un uso muy común de esta técnica se practica cuando los servidores de distribución de Linux alojan imágenes de CD o DVD descargables de su software

junto con archivos que contienen los valores de suma de comprobación calculados de esas imágenes de disco. Aquí hay un listado de ejemplo de un espejo de descarga de Debian:

```
□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□
[PARENTDIR] Parent Directory                                      -
[SUM]       MD5SUMS                              2019-09-08 17:46 274
[CRT]       MD5SUMS.sign                         2019-09-08 17:52 833
[SUM]       SHA1SUMS                             2019-09-08 17:46 306
[CRT]       SHA1SUMS.sign                        2019-09-08 17:52 833
[SUM]       SHA256SUMS                           2019-09-08 17:46 402
[CRT]       SHA256SUMS.sign                      2019-09-08 17:52 833
[SUM]       SHA512SUMS                           2019-09-08 17:46 658
[CRT]       SHA512SUMS.sign                      2019-09-08 17:52 833
[ISO]       debian-10.1.0-amd64-netinst.iso      2019-09-08 04:37 335M
[ISO]       debian-10.1.0-amd64-xfce-CD-1.iso    2019-09-08 04:38 641M
[ISO]       debian-edu-10.1.0-amd64-netinst.iso  2019-09-08 04:38 405M
[ISO]       debian-mac-10.1.0-amd64-netinst.iso  2019-09-08 04:38 334M
□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□
```

En la lista anterior, los archivos de imagen del instalador de Debian están acompañados por archivos de texto que contienen sumas de verificación de los archivos de los diversos algoritmos (MD5, SHA1, SHA256 y SHA512).

> **NOTE** Una suma de comprobación es un valor derivado de un cálculo matemático, basado en una función hash criptográfica, contra un archivo. Existen diferentes tipos de funciones hash criptográficas que varían en intensidad. El examen esperará que esté familiarizado con el uso de `md5sum`, `sha256sum` y `sha512sum`.

Una vez que descargue un archivo (por ejemplo, la imagen `debian-10.1.0-amd64-netinst.iso`), comparará la suma de comprobación del archivo que se descargó con un valor de suma de comprobación que se le proporcionó.

Aquí hay un ejemplo para ilustrar el punto. Calcularemos el valor SHA256 del archivo `ftu.txt` utilizando el comando `sha256sum`:

```
$ sha256sum ftu.txt
345452304fc26999a715652543c352e5fc7ee0c1b9deac6f57542ec91daf261c  ftu.txt
```

La larga cadena de caracteres que precede al nombre del archivo es el valor de suma de comprobación SHA256 de este archivo de texto. Cree un archivo que contenga ese valor, para usarlo

durante la verificación de la integridad de nuestro archivo de texto original. Podemos hacer esto con el mismo comando `sha256sum` y redirigir la salida a un archivo:

```
$ sha256sum ftu.txt > sha256.txt
```

Ahora, para verificar el archivo `ftu.txt`, solo usamos el mismo comando y proporcionamos el nombre de archivo que contiene nuestro valor de suma de comprobación junto con `-c`:

```
$ sha256sum -c sha256.txt
ftu.txt: OK
```

El valor contenido en el archivo coincide con la suma de comprobación SHA256 calculada para nuestro archivo `ftu.txt`, tal como se espera. Sin embargo, si se modificara el archivo original (como algunos bytes perdidos durante la descarga de un archivo, o alguien lo hubiera alterado deliberadamente), la comprobación del valor fallará. En tales casos, sabemos que nuestro archivo está dañado o corrupto y no podemos confiar en la integridad de su contenido. Para probar el punto, agregaremos texto al final del archivo:

```
$ echo "new entry" >> ftu.txt
```

Ahora intentaremos verificar la integridad del archivo:

```
$ sha256sum -c sha256.txt
ftu.txt: FAILED
sha256sum: WARNING: 1 computed checksum did NOT match
```

Y vemos que la suma de comprobación no coincide con lo que se esperaba para el archivo. Por lo tanto, no podemos confiar en la integridad de este archivo. Podríamos intentar descargar una nueva copia de un archivo, informar el fallo de la suma de verificación al remitente del archivo o informarlo al equipo de seguridad del centro de datos, según la importancia del archivo.

Buscando más en los archivos

El comando octal dump (`od`) a menudo se usa para depurar aplicaciones y varios archivos. Por sí solo, el comando `od` solo enumerará el contenido de un archivo en formato octal. Podemos usar nuestro archivo `ftu.txt` de antes para practicar con este comando:

```
$ od ftu.txt
0000000 075142 060543 005164 060543 005164 072543 005164 062550
0000020 062141 066012 071545 005163 062155 071465 066565 067012
0000040 005154 062157 070012 071541 062564 071412 062145 071412
0000060 060550 032462 071466 066565 071412 060550 030465 071462
0000100 066565 071412 071157 005164 070163 064554 005164 060564
0000120 066151 072012 005162 067165 070551 073412 005143 075170
0000140 060543 005164 061572 072141 000012
0000151
```

La primera columna de salida es el desplazamiento de bytes para cada línea de salida. Como `od` imprime la información en formato octal de manera predeterminada, cada línea comienza con el desplazamiento de bytes de ocho bits, seguido de ocho columnas, cada una con el valor octal de los datos dentro de esa columna.

TIP Recuerde que un *byte* tiene 8 bits de longitud.

Si necesita ver el contenido de un archivo en formato hexadecimal, use la opción `-x`:

```
$ od -x ftu.txt
0000000 7a62 6163 0a74 6163 0a74 7563 0a74 6568
0000020 6461 6c0a 7365 0a73 646d 7335 6d75 6e0a
0000040 0a6c 646f 700a 7361 6574 730a 6465 730a
0000060 6168 3532 7336 6d75 730a 6168 3135 7332
0000100 6d75 730a 726f 0a74 7073 696c 0a74 6174
0000120 6c69 740a 0a72 6e75 7169 770a 0a63 7a78
0000140 6163 0a74 637a 7461 000a
0000151
```

Ahora cada una de las ocho columnas después del desplazamiento de bytes está representada por sus equivalentes hexadecimales.

Un uso útil del comando `od` es para depurar scripts. Por ejemplo, el comando `od` puede mostrarnos caracteres que normalmente no se ven que existen dentro de un archivo, como las entradas *newline*. Podemos hacer esto con la opción `-c`, de modo que en lugar de mostrar la notación numérica para cada byte, estas entradas de columna se mostrarán como sus equivalentes de caracteres:

```
$ od -c ftu.txt
0000000   b   z   c   a   t  \n   c   a   t  \n   c   u   t  \n   h   e
0000020   a   d  \n   l   e   s   s  \n   m   d   5   s   u   m  \n   n
0000040   l  \n   o   d  \n   p   a   s   t   e  \n   s   e   d  \n   s
0000060   h   a   2   5   6   s   u   m  \n   s   h   a   5   1   2   s
0000100   u   m  \n   s   o   r   t  \n   s   p   l   i   t  \n   t   a
0000120   i   l  \n   t   r  \n   u   n   i   q  \n   w   c  \n   x   z
0000140   c   a   t  \n   z   c   a   t  \n
0000151
```

Todas las entradas de nueva línea dentro del archivo están representadas por los caracteres ocultos `\n`. Si solo desea ver todos los caracteres dentro de un archivo y no necesita ver la información de desplazamiento de bytes, la columna de desplazamiento de bytes se puede eliminar de la salida de la siguiente manera:

```
$ od -An -c ftu.txt
   b   z   c   a   t  \n   c   a   t  \n   c   u   t  \n   h   e
   a   d  \n   l   e   s   s  \n   m   d   5   s   u   m  \n   n
   l  \n   o   d  \n   p   a   s   t   e  \n   s   e   d  \n   s
   h   a   2   5   6   s   u   m  \n   s   h   a   5   1   2   s
   u   m  \n   s   o   r   t  \n   s   p   l   i   t  \n   t   a
   i   l  \n   t   r  \n   u   n   i   q  \n   w   c  \n   x   z
   c   a   t  \n   z   c   a   t  \n
```

Ejercicios Guiados

1. Alguien acaba de donar una computadora portátil a su escuela y ahora desea instalar Linux en ella. No hay manual y se vio obligado a arrancarlo desde una unidad de memoria USB sin gráficos en absoluto. Obtiene acceso a un terminal de shell y se conoce, para cada procesador que tenga, habrá una línea para él en el archivo `/proc/cpuinfo`:

```
processor   : 0
vendor_id   : GenuineIntel
cpu family  : 6
model       : 158

(líneas saltadas)

processor   : 1
vendor_id   : GenuineIntel
cpu family  : 6
model       : 158

(se saltaron más líneas)
```

   - Usando los comandos `grep` y `wc` muestre cuántos procesadores tiene.

   - Haga lo mismo con `sed` en lugar de `grep`.

2. Explore su archivo local `/etc/passwd` con los comandos `grep`, `sed`, `head` y `tail` según las siguientes tareas:

   - ¿Qué usuarios tienen acceso a un shell Bash?

   - Su sistema tiene varios usuarios que existen para manejar programas específicos o para fines administrativos. No tienen acceso a un shell. ¿Cuántos de esos existen en su sistema?

   - ¿Cuántos usuarios y grupos existen en su sistema (recuerde: use solo el archivo `/etc/passwd`)?

   ◦ Muestre solo la primera línea, la última línea y la décima línea de su archivo `/etc/passwd`.

3. Considere este ejemplo de archivo `/etc/passwd`. Copie las líneas siguientes en un archivo local llamado `mypasswd` para este ejercicio.

```
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
nvidia-persistenced:x:121:128:NVIDIA Persistence
Daemon,,,:/nonexistent:/sbin/nologin
libvirt-qemu:x:64055:130:Libvirt Qemu,,,:/var/lib/libvirt:/usr/sbin/nologin
libvirt-dnsmasq:x:122:133:Libvirt
Dnsmasq,,,:/var/lib/libvirt/dnsmasq:/usr/sbin/nologin
carol:x:1000:2000:Carol Smith,Finance,,,Main Office:/home/carol:/bin/bash
dave:x:1001:1000:Dave Edwards,Finance,,,Main Office:/home/dave:/bin/ksh
emma:x:1002:1000:Emma Jones,Finance,,,Main Office:/home/emma:/bin/bash
frank:x:1003:1000:Frank Cassidy,Finance,,,Main Office:/home/frank:/bin/bash
grace:x:1004:1000:Grace Kearns,Engineering,,,Main Office:/home/grace:/bin/ksh
henry:x:1005:1000:Henry Adams,Sales,,,Main Office:/home/henry:/bin/bash
john:x:1006:1000:John Chapel,Sales,,,Main Office:/home/john:/bin/bash
```

   ◦ Liste todos los usuarios en el grupo `1000` (use `sed` para seleccionar solo el campo apropiado) de su archivo `mypasswd`.

   ◦ Liste solo los nombres completos de todos los usuarios de este grupo (use `sed` y `cut`).

Ejercicios Exploratorios

1. Una vez más, utilizando el archivo `mypasswd` de los ejercicios anteriores, idee un comando Bash que seleccionará a una persona de la Oficina Principal para ganar un concurso de rifas. Use el comando `sed` para imprimir solo las líneas de la Oficina Principal, y luego una secuencia de comando `cut` para recuperar el nombre de cada usuario de estas líneas. A continuación, deseará ordenar aleatoriamente estos nombres y solo imprimir el nombre superior de la lista.

2. ¿Cuántas personas trabajan en Finanzas, Ingeniería y Ventas? (Considere explorar el comando `uniq`).

3. Ahora desea preparar un archivo CSV (valores separados por comas) para poder importar fácilmente, desde el archivo `mypasswd` del ejemplo anterior, el archivo `names.csv` a LibreOffice. El contenido del archivo tendrá el siguiente formato:

```
First Name,Last Name,Position
Carol,Smith,Finance
...
John,Chapel,Sales
```

   Consejo: Utilice los comandos `sed`, `cut` y `paste` para lograr los resultados deseados. Tenga en cuenta que la coma (`,`) será el delimitador de este archivo.

4. Supongamos que la hoja de cálculo `names.csv` creada en el ejercicio anterior es un archivo importante y queremos asegurarnos de que nadie lo manipulará desde el momento en que se lo enviamos a alguien y el momento en que nuestro destinatario lo recibe. ¿Cómo podemos asegurar la integridad de este archivo usando `md5sum`?

5. Te prometiste a ti mismo que leerías un libro clásico de 100 líneas por día y decidiste comenzar con Mariner y Mystic de Herman Melville. Diseñe un comando usando `split` que separe este libro en secciones de 100 líneas cada una. Para obtener el libro en formato de texto plano, búsquelo en https://www.gutenberg.org.

6. Usando `ls -l` en el directorio `/etc`, ¿qué tipo de listado obtiene? Usando el comando `cut` en la salida del comando `ls` dado, ¿cómo mostraría solo los nombres de archivo? ¿Qué pasa con el

nombre del archivo y el propietario de los archivos? Junto con los comandos `ls -l` y `cut`, utilice el comando `tr` para *suprimir* las apariciones múltiples de un espacio en un solo espacio para ayudar a formatear la salida con un comando `cut`.

7. Este ejercicio asume que está en una máquina real (no en una máquina virtual). También debe tener una memoria USB con usted. Revise las páginas del manual para el comando `tail` y descubra cómo seguir un archivo a medida que se le agrega texto. Mientras monitorea la salida de un comando `tail` en el archivo `/var/log/syslog`, inserte una memoria USB. Escriba el comando completo que usaría para obtener el Producto, el Fabricante y la cantidad total de almacenamiento de su memoria USB.

Resumen

Tratar con flujos de texto es de gran importancia cuando se administra cualquier sistema Linux. Las secuencias de texto se pueden procesar utilizando scripts para automatizar las tareas diarias o encontrar información de depuración relevante en los archivos de registro. Aquí hay un breve resumen de los comandos cubiertos en esta lección:

`cat`
: Se usa para combinar o leer archivos de texto sin formato.

`bzcat`
: Permite el procesamiento o lectura de archivos comprimidos utilizando el método `bzip2`.

`xzcat`
: Permite el procesamiento o la lectura de archivos comprimidos utilizando el método `xz`.

`zcat`
: Permite el procesamiento o la lectura de archivos comprimidos utilizando el método `xz`.

`less`
: Este comando pagina los contenidos de un archivo, y permite la navegación y la funcionalidad de búsqueda.

`head`
: Este comando mostrará las primeras 10 líneas de un archivo de forma predeterminada. Con el uso de la opción `-n` se pueden mostrar menos o más líneas.

`tail`
: Este comando mostrará las últimas 10 líneas de un archivo de forma predeterminada. Con el uso de la opción `-n` se pueden mostrar menos o más líneas. La opción `-f` se utiliza para seguir la salida de un archivo de texto cuando se escriben nuevos datos.

`wc`
: Abreviatura de “word count”, pero dependiendo de los parámetros que use contará caracteres, palabras y líneas.

`sort`
: Se utiliza para organizar la salida de una lista alfabéticamente, alfabéticamente inversa o en orden aleatorio.

`uniq`
: Se usa para enumerar (y contar) cadenas coincidentes.

`od`
: El comando "octal dump" se utiliza para mostrar un archivo binario en notación octal, decimal o hexadecimal.

`nl`
: El comando "number line" mostrará el número de líneas en un archivo, así como volver a crear un archivo con cada línea precedida por su número de línea.

`sed`
: El editor de flujo se puede usar para encontrar ocurrencias coincidentes de cadenas usando Expresiones regulares, así como editar archivos usando patrones predefinidos.

`tr`
: El comando traducir puede reemplazar caracteres y también elimina y comprime caracteres repetidos.

`cut`
: Este comando puede imprimir columnas de archivos de texto como campos basados en el delimitador de caracteres de un archivo.

`paste`
: Unir archivos en columnas según el uso de separadores de campo.

`split`
: Este comando puede dividir archivos más grandes en archivos más pequeños según los criterios establecidos por las opciones del comando.

`md5sum`
: Se utiliza para calcular el valor hash MD5 de un archivo. También se utiliza para verificar un archivo contra un valor hash existente para garantizar la integridad de un archivo.

`sha256sum`
: Se utiliza para calcular el valor hash SHA256 de un archivo. También se utiliza para verificar un archivo contra un valor hash existente para garantizar la integridad de un archivo.

`sha512sum`

Se utiliza para calcular el valor hash SHA512 de un archivo. También se utiliza para verificar un archivo contra un valor hash existente para garantizar la integridad de un archivo.

Respuestas a los ejercicios guiados

1. Alguien acaba de donar una computadora portátil a su escuela y ahora desea instalar Linux en ella. No hay manual y se vio obligado a arrancarlo desde una unidad de memoria USB sin gráficos en absoluto. Obtiene acceso a un terminal de shell y se conoce, para cada procesador que tenga, habrá una línea para él en el archivo `/proc/cpuinfo`:

```
processor   : 0
vendor_id   : GenuineIntel
cpu family  : 6
model       : 158

(líneas saltadas)

processor   : 1
vendor_id   : GenuineIntel
cpu family  : 6
model       : 158

(se saltaron más líneas)
```

   - Usando los comandos `grep` y `wc` muestra cuántos procesadores tiene.

     Aquí hay dos opciones:

```
$ cat /proc/cpuinfo | grep processor | wc -l
$ grep processor /proc/cpuinfo | wc -l
```

     Ahora que sabe que hay varias maneras en que puede hacer lo mismo, ¿cuándo debería usar una u otra? Realmente depende de varios factores, los dos más importantes son el rendimiento y la legibilidad. La mayoría de las veces usará comandos de shell dentro de scripts de shell para automatizar sus tareas y cuanto más grandes y complejos sean sus scripts, más tendrá que preocuparse por mantenerlos rápidos.

   - Haga lo mismo con `sed` en lugar de `grep`

     Ahora, en lugar de `grep` intentaremos esto con `sed`:

```
$ sed -n /processor/p /proc/cpuinfo | wc -l
```

Aquí usamos `sed` con el parámetro `-n` para que `sed` no imprima nada excepto lo que coincide con la expresión `processor`, como lo indica el comando `p`. Como hicimos en las soluciones `grep`, `wc -l` contará la cantidad de líneas, por lo tanto, la cantidad de procesadores que tenemos.

Estudie el siguiente ejemplo:

```
$ sed -n /processor/p /proc/cpuinfo | sed -n '$='
```

Esta secuencia de comandos proporciona resultados idénticos al ejemplo anterior donde la salida de `sed` se canalizaba a un comando `wc`. La diferencia aquí es que, en lugar de usar `wc -l` para contar el número de líneas, se invoca nuevamente `sed` para proporcionar una funcionalidad equivalente. Una vez más, estamos suprimiendo la salida de `sed` con la opción `-n`, excepto por la expresión que llamamos explícitamente, que es `'$='`. Esta expresión le dice a `sed` que coincida con la última línea (`$`) y luego imprima ese número de línea (`=`).

2. Explore su archivo local `/etc/passwd` con los comandos `grep`, `sed`, `head` y `tail` según las siguientes tareas:

   - ¿Qué usuarios tienen acceso a un shell Bash?

     ```
     $ grep ":/bin/bash$" /etc/passwd
     ```

     Mejoraremos esta respuesta mostrando solo el nombre del usuario que utiliza el shell Bash.

     ```
     $ grep ":/bin/bash$" /etc/passwd | cut -d: -f1
     ```

     El nombre de usuario es el primer campo (parámetro `-f1` del comando `cut`) y el archivo `/etc/passwd` usa `:` como separadores (parámetro `-d:` del comando `cut`) simplemente canalice la salida del comando `grep` al comando `cut` apropiado.

   - Su sistema tiene varios usuarios que existen para manejar programas específicos o para fines administrativos. No tienen acceso a un shell. ¿Cuántos de esos existen en su sistema?

     La forma más fácil de encontrar esto es imprimiendo las líneas de las cuentas que no usan el shell Bash:

     ```
     $ grep -v ":/bin/bash$" /etc/passwd | wc -l
     ```

- ¿Cuántos usuarios y grupos existen en su sistema? (recuerde: use solo el archivo `/etc/passwd`)

  El primer campo de cualquier línea dada en su archivo `/etc/passwd` es el nombre de usuario, el segundo es típicamente una `x` que indica que la contraseña del usuario no está almacenada aquí (está encriptada en el archivo `/etc/shadow`) El tercero es el ID de usuario (UID) y el cuarto es el ID de grupo (GID). Entonces esto debería darnos la cantidad de usuarios:

  ```
  $ cut -d: -f3 /etc/passwd | wc -l
  ```

  Bueno, la mayoría de las veces lo hará. Sin embargo, hay situaciones en las que establecerá diferentes superusuarios u otros tipos especiales de usuarios que comparten el mismo UID (ID de usuario). Entonces, para estar seguros, canalizaremos el resultado de nuestro comando `cut` al comando `sort` y luego contaremos el número de líneas.

  ```
  $ cut -d: -f3 /etc/passwd | sort -u | wc -l
  ```

  Ahora, para el número de grupos:

  ```
  $ cut -d: -f4 /etc/passwd | sort -u | wc -l
  ```

- Muestre solo la primera línea, la última línea y la décima línea de su archivo `/etc/passwd`

  Esto lo hará:

  ```
  $ sed -n -e '1'p -e '10'p -e '$'p /etc/passwd
  ```

  Recuerde que el parámetro `-n` le dice a `sed` que no imprima nada más que lo especificado por el comando `p`. El signo de dólar ($) usado aquí es una expresión regular que significa la última línea del archivo.

3. Considere este ejemplo de archivo `/etc/passwd`. Copie las líneas siguientes en un archivo local llamado `mypasswd` para este ejercicio.


```
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
nvidia-persistenced:x:121:128:NVIDIA Persistence
Daemon,,,:/nonexistent:/sbin/nologin
libvirt-qemu:x:64055:130:Libvirt Qemu,,,:/var/lib/libvirt:/usr/sbin/nologin
libvirt-dnsmasq:x:122:133:Libvirt
Dnsmasq,,,:/var/lib/libvirt/dnsmasq:/usr/sbin/nologin
carol:x:1000:2000:Carol Smith,Finance,,,Main Office:/home/carol:/bin/bash
dave:x:1001:1000:Dave Edwards,Finance,,,Main Office:/home/dave:/bin/ksh
emma:x:1002:1000:Emma Jones,Finance,,,Main Office:/home/emma:/bin/bash
frank:x:1003:1000:Frank Cassidy,Finance,,,Main Office:/home/frank:/bin/bash
grace:x:1004:1000:Grace Kearns,Engineering,,,Main Office:/home/grace:/bin/ksh
henry:x:1005:1000:Henry Adams,Sales,,,Main Office:/home/henry:/bin/bash
john:x:1006:1000:John Chapel,Sales,,,Main Office:/home/john:/bin/bash
```

- Liste todos los usuarios en el grupo 1000 (use `sed` para seleccionar solo el campo apropiado) de su archivo `mypasswd`:

  El GID es el cuarto campo en el archivo `/etc/passwd`. Puede tener la tentación de probar esto:

  ```
  $ sed -n /1000/p mypasswd
  ```

  En este caso, también obtendrá esta línea:

  ```
  carol:x:1000:2000:Carol Smith,Finance,,,Main Office:/home/carol:/bin/bash
  ```

  Esto no es correcto ya que Carol Smith es miembro de GID 2000 y la coincidencia se produjo debido al UID. Sin embargo, es posible que haya notado que después del GID, el siguiente campo comienza con un carácter en mayúscula. Podemos usar una expresión regular para resolver este problema.

  ```
  $ sed -n /:1000:[A-Z]/p mypasswd
  ```

  La expresión `[A-Z]` coincidirá con cualquier carácter en mayúsculas. Aprenderá más sobre esto en la lección respectiva.

- Liste solo los nombres completos de todos los usuarios de este grupo (use `sed` y `cut`):

  Utilice la misma técnica que utilizó para resolver la primera parte de este ejercicio y canalícela a un comando de corte.

  ```
  $ sed -n /:1000:[A-Z]/p mypasswd | cut -d: -f5
  Dave Edwards,Finance,,,Main Office
  Emma Jones,Finance,,,Main Office
  Frank Cassidy,Finance,,,Main Office
  Grace Kearns,Engineering,,,Main Office
  Henry Adams,Sales,,,Main Office
  John Chapel,Sales,,,Main Office
  ```

  ¡No del todo! Tenga en cuenta cómo los campos dentro de sus resultados pueden estar separados por `,`. Así que canalizaremos la salida a otro comando `cut`, usando `,` como delimitador.

  ```
  $ sed -n /:1000:[A-Z]/p mypasswd | cut -d: -f5 | cut -d, -f1
  Dave Edwards
  Emma Jones
  Frank Cassidy
  Grace Kearns
  Henry Adams
  John Chapel
  ```

Respuestas a ejercicios exploratorios

1. Una vez más, utilizando el archivo `mypasswd` de los ejercicios anteriores, elabore un comando Bash que seleccione a una persona de la oficina principal para ganar un concurso de rifa. Utilice el comando `sed` para imprimir solo las líneas para la oficina principal, y luego una secuencia de comando `cut` para recuperar el nombre de cada usuario de estas líneas. A continuación, querrá ordenar estos nombres al azar e imprimir solo el nombre superior de la lista.

   Primero explore cómo el parámetro `-R` manipula la salida del comando `sort`. Repita este comando un par de veces en su máquina (tenga en cuenta que deberá encerrar `Main Office` entre comillas simples, por lo que `sed` lo manejará como una sola cadena):

   ```
   $ sed -n /'Main Office'/p mypasswd | cut -d: -f5 | cut -d, -f1 | sort -R
   ```

   Aquí hay una solución al problema:

   ```
   $ sed -n /'Main Office'/p mypasswd | cut -d: -f5 | cut -d, -f1 | sort -R | head
   -1
   ```

2. ¿Cuántas personas trabajan en Finanzas, Ingeniería y Ventas? (Considere explorar el comando `uniq`).

   Siga construyendo sobre la base de lo que aprendió en los ejercicios anteriores. Intente lo siguiente:

   ```
   $ sed -n /'Main Office'/p mypasswd
   $ sed -n /'Main Office'/p mypasswd | cut -d, -f2
   ```

   Observe que ahora no nos importa el `:` como delimitador. Solo queremos el segundo campo cuando dividimos las líneas por los caracteres `,`.

   ```
   $ sed -n /'Main Office'/p mypasswd | cut -d, -f2 | uniq -c
         4 Finance
         1 Engineering
         2 Sales
   ```

   El comando `uniq` solo generará las líneas únicas (no las líneas repetidas) y el parámetro `-c` le dice a `uniq` que cuente las ocurrencias de las líneas iguales. Aquí hay una advertencia: `uniq` solo

considerará líneas adyacentes. Cuando este no sea el caso, tendrá que usar el comando `sort`.

3. Ahora desea preparar un archivo CSV (valores separados por comas) para que pueda importar fácilmente, desde el archivo `mypasswd` en el ejemplo anterior, el archivo `names.csv` a LibreOffice. El contenido del archivo tendrá el siguiente formato:

```
First Name,Last Name,Position
Carol,Smith,Finance
...
John,Chapel,Sales
```

Tip: Utilice los comandos `sed`, `cut` y `paste` para lograr los resultados deseados. Tenga en cuenta que la coma (,) será el delimitador de este archivo.

Comience con los comandos `sed` y `cut`, basándose en lo que aprendimos de los ejercicios anteriores:

```
$ sed -n /'Main Office'/p mypasswd | cut -d: -f5 | cut -d" " -f1 > firstname
```

Ahora tenemos el archivo `firstname` con los nombres de nuestros empleados.

```
$ sed -n /'Main Office'/p mypasswd | cut -d: -f5 | cut -d" " -f2 | cut -d, -f1
> lastname
```

Ahora tenemos el archivo `lastname` que contiene los apellidos de cada empleado.

A continuación, determinamos en qué departamento trabaja cada empleado:

```
$ sed -n /'Main Office'/p mypasswd | cut -d: -f5 | cut -d, -f2 > department
```

Antes de trabajar en la solución final, pruebe los siguientes comandos para ver qué tipo de salida generan:

```
$ cat firstname lastname department
$ paste firstname lastname department
```

Y ahora para la solución final:

```
$ paste firstname lastname department | tr '\t' ,
$ paste firstname lastname department | tr '\t' , > names.csv
```

Aquí usamos el comando `tr` para *traducir* `\t`, el separador de tabulación, por un `,`. `tr` es bastante útil cuando necesitamos intercambiar un carácter por otro. Asegúrese de revisar las páginas de manual para `tr` y `paste`. Por ejemplo, podemos usar la opción `-d` para el delimitador para hacer que el comando anterior sea menos complejo:

```
$ paste -d, firstname lastname department
```

Usamos el comando `paste` aquí una vez que necesitábamos familiarizarte con él. Sin embargo, podríamos haber realizado fácilmente todas las tareas en una sola cadena de comando:

```
$ sed -n /'Main Office'/p mypasswd | cut -d: -f5 | cut -d, -f1,2 | tr ' ' , >
names.csv
```

4. Supongamos que la hoja de cálculo `names.csv` creada en el ejercicio anterior es un archivo importante y queremos asegurarnos de que nadie lo manipulará desde el momento en que se lo enviamos a alguien y el momento en que nuestro destinatario lo recibe. ¿Cómo podemos asegurar la integridad de este archivo usando `md5sum`?

   Si busca en las páginas de manual `md5sum`, `sha256sum` y `sha512sum`, verá que todas comienzan con el siguiente texto:

   “compute and check XXX message digest”

   Donde “XXX” es el algoritmo que se utilizará para crear este *resumen de mensajes*.

   Usaremos `md5sum` como ejemplo y luego puedes probar con los otros comandos.

```
$ md5sum names.csv
61f0251fcab61d9575b1d0cbf0195e25  names.csv
```

   Ahora, por ejemplo, puede hacer que el archivo esté disponible a través de un servicio ftp seguro y enviar el *mensaje resumen* generado utilizando otro medio seguro de comunicación. Si el archivo ha sido ligeramente modificado, el *mensaje resumen* será completamente diferente. Solo para probarlo, edite `names.csv` y cambie Jones a James como se muestra aquí:

```
$ sed -i.backup s/Jones/James/ names.csv
$ md5sum names.csv
f44a0d68cb480466099021bf6d6d2e65  names.csv
```

Siempre que haga que los archivos estén disponibles para descargar, también es una buena práctica distribuir un *message digest* para que las personas que descarguen su archivo puedan producir un nuevo *message digest* y compararlo con el original. Si navega a través de https://kernel.org encontrará la página https://mirrors.edge.kernel.org/pub/linux/kernel/v5.x/sha256sums.asc donde puede obtener el sha256sum para todos los archivos disponible para descarga.

5. Te prometiste a ti mismo que leerías un libro clásico de 100 líneas por día y decidiste comenzar con Mariner y Mystic de Herman Melville. Diseñe un comando usando `split` que separe este libro en secciones de 100 líneas cada una. Para obtener el libro en formato de texto plano, búsquelo en https://www.gutenberg.org.

   Primero obtendremos el libro completo del sitio del Proyecto Gutenberg, donde puede obtener este y otros libros que están disponibles en el dominio público.

   ```
   $ wget https://www.gutenberg.org/files/50461/50461-0.txt
   ```

   Es posible que necesite instalar `wget` si aún no está instalado en su sistema. Alternativamente, también puede usar `curl`. Use `less` para verificar el libro:

   ```
   $ less 50461-0.txt
   ```

   Ahora dividiremos el libro en trozos de 100 líneas cada uno:

   ```
   $ split -l 100 -d 50461-0.txt melville
   ```

   `50461-0.txt` es el archivo que dividiremos. `melville` será el prefijo para los archivos divididos. El `-l 100` especifica el número de líneas y la opción `-d` le dice a `split` que numere los archivos (usando el sufijo proporcionado). Puede usar `nl` en cualquiera de los archivos divididos (probablemente no en el último) y confirmar que cada uno de ellos tenga 100 líneas.

6. Usando `ls -l` en el directorio `/etc`, ¿qué tipo de listado obtiene? Usando el comando `cut` en la salida del comando `ls` dado, ¿cómo mostraría solo los nombres de archivo? ¿Qué pasa con el nombre del archivo y el propietario de los archivos? Junto con los comandos `ls -l` y `cut`, utilice

el comando `tr` para *suprimir* las apariciones múltiples de un espacio en un solo espacio para ayudar a formatear la salida con un comando `cut`.

El comando `ls` por sí solo le dará solo los nombres de los archivos. Sin embargo, podemos preparar la salida de `ls -l` (la lista larga) para extraer información más específica.

```
$ ls -l /etc | tr -s ' ' ,
drwxr-xr-x,3,root,root,4096,out,24,16:58,acpi
-rw-r--r--,1,root,root,3028,dez,17,2018,adduser.conf
-rw-r--r--,1,root,root,10,out,2,17:38,adjtime
drwxr-xr-x,2,root,root,12288,out,31,09:40,alternatives
-rw-r--r--,1,root,root,401,mai,29,2017,anacrontab
-rw-r--r--,1,root,root,433,out,1,2017,apg.conf
drwxr-xr-x,6,root,root,4096,dez,17,2018,apm
drwxr-xr-x,3,root,root,4096,out,24,16:58,apparmor
drwxr-xr-x,9,root,root,4096,nov,6,20:20,apparmor.d
```

El parámetro `-s` indica a `tr` que reduzca los espacios repetidos en una sola instancia de un espacio. El comando `tr` funciona para cualquier tipo de carácter repetitivo que especifique. Luego reemplazamos los espacios con una coma `,`. En realidad, no necesitamos reemplazar los espacios en nuestro ejemplo, así que simplemente omitiremos el `,`.

```
$ ls -l /etc | tr -s ' '
drwxr-xr-x 3 root root 4096 out 24 16:58 acpi
-rw-r--r-- 1 root root 3028 dez 17 2018 adduser.conf
-rw-r--r-- 1 root root 10 out 2 17:38 adjtime
drwxr-xr-x 2 root root 12288 out 31 09:40 alternatives
-rw-r--r-- 1 root root 401 mai 29 2017 anacrontab
-rw-r--r-- 1 root root 433 out 1 2017 apg.conf
drwxr-xr-x 6 root root 4096 dez 17 2018 apm
drwxr-xr-x 3 root root 4096 out 24 16:58 apparmor
```

Si solo se desean los nombres de archivo, todo lo que necesitamos que se muestre es el noveno campo:

```
$ ls -l /etc | tr -s ' ' | cut -d" " -f9
```

Para el nombre de archivo y el propietario de un archivo necesitaremos los campos noveno y tercero:

```
$ ls -l /etc | tr -s ' ' | cut -d" " -f9,3
```

¿Qué pasa si solo necesitamos los nombres de las carpetas y su propietario?

```
$ ls -l /etc | grep ^d | tr -s ' ' | cut -d" " -f9,3
```

7. Este ejercicio asume que está en una máquina real (no en una máquina virtual). También debe tener una memoria USB con usted. Revise las páginas del manual para el comando `tail` y descubra cómo seguir un archivo a medida que se le agrega texto. Mientras monitorea la salida de un comando `tail` en el archivo `/var/log/syslog`, inserte una memoria USB. Escriba el comando completo que usaría para obtener el Producto, el Fabricante y la cantidad total de almacenamiento de su memoria USB.

```
$ tail -f /var/log/syslog | grep -i 'product\:\|blocks\|manufacturer'
Nov  8 06:01:35 brod-avell kernel: [124954.369361] usb 1-4.3: Product: Cruzer
Blade
Nov  8 06:01:35 brod-avell kernel: [124954.369364] usb 1-4.3: Manufacturer:
SanDisk
Nov  8 06:01:37 brod-avell kernel: [124955.419267] sd 2:0:0:0: [sdc] 61056064
512-byte logical blocks: (31.3 GB/29.1 GiB)
```

Por supuesto, este es un ejemplo y los resultados pueden variar según el fabricante de la memoria USB. Observe que ahora usamos el parámetro `-i` con el comando `grep` ya que no estamos seguros de si las cadenas que estamos buscando estaban en mayúsculas o minúsculas. También utilizamos `|` como un OR lógico, por lo que buscamos líneas que contengan `product` OR `blocks` OR `manufacturer`.



103.3 Administración básica de archivos

Referencia al objetivo del LPI

LPIC-1 v5, Exam 101, Objective 103.3

Importancia

4

Áreas de conocimiento clave

- Copiar, mover y eliminar archivos y directorios de forma individual.
- Copiar múltiples archivos y directorios de forma recursiva.
- Eliminar archivos y directorios de forma recursiva.
- Utilizar especificaciones de comodines simples y avanzadas en los comandos.
- Usar find para localizar archivos y actuar sobre ellos en base a su tipo, tamaño o marcas de tiempo.
- Uso de tar, cpio y dd.

Lista parcial de archivos, términos y utilidades

- `cp`
- `find`
- `mkdir`
- `mv`
- `ls`
- `rm`
- `rmdir`

- `touch`
- `tar`
- `cpio`
- `dd`
- `file`
- `gzip`
- `gunzip`
- `bzip2`
- `bunzip2`
- Uso de comodines (file globbing)



103.3 Lección 1

Certificación:	LPIC-1
Versión:	5.0
Tema:	103 Comandos GNU y Unix
Objetivo:	103.3 Gestión básica de archivos
Lección:	1 de 2

Introducción

Todo en Linux es un archivo, por lo que saber cómo manipularlos es muy importante. En esta lección, cubriremos operaciones básicas en archivos.

En general, como usuario de Linux, se le pedirá que navegue por el sistema de archivos, copie archivos de una ubicación a otra y elimine archivos. También cubriremos los comandos asociados con la gestión de archivos.

Un archivo es una entidad que almacena datos y programas. Se compone de contenido y metadatos (tamaño del archivo, propietario, fecha de creación, permisos). Los archivos están organizados en directorios. Un directorio es un archivo que almacena otros archivos.

Los diferentes tipos de archivos incluyen:

Archivos regulares

que almacenan datos y programas.

Directorios

que contienen otros archivos.

Archivos especiales

que se utilizan para entrada y salida.

Por supuesto, existen otros tipos de archivos, pero están más allá del alcance de esta lección. Más adelante, discutiremos cómo identificar estos diferentes tipos de archivos.

Manipulación de Archivos

Usando `ls` para listar archivos

El comando `ls` es una de las herramientas de línea de comandos más importantes que debe aprender para navegar por el sistema de archivos.

En su forma básica, `ls` listará los nombres de archivos y directorios *solamente*:

```
$ ls
Desktop Downloads   emp_salary  file1   Music   Public  Videos
Documents   emp_name    examples.desktop    file2   Pictures    Templates
```

Cuando se usa con `-l`, denominado formato de "lista larga", muestra los permisos de archivo o directorio, propietario, tamaño, fecha de modificación, hora y nombre:

```
$ ls -l
total 60
drwxr-xr-x  2   frank   frank   4096    Apr 1   2018    Desktop
drwxr-xr-x  2   frank   frank   4096    Apr 1   2018    Documents
drwxr-xr-x  2   frank   frank   4096    Apr 1   2018    Downloads
-rw-r--r--  1   frank   frank     21    Sep 7   12:59   emp_name
-rw-r--r--  1   frank   frank     20    Sep 7   13:03   emp_salary
-rw-r--r--  1   frank   frank   8980    Apr 1   2018    examples.desktop
-rw-r--r--  1   frank   frank     10    Sep 1   2018    file1
-rw-r--r--  1   frank   frank     10    Sep 1   2018    file2
drwxr-xr-x  2   frank   frank   4096    Apr 1   2018    Music
drwxr-xr-x  2   frank   frank   4096    Apr 1   2018    Pictures
drwxr-xr-x  2   frank   frank   4096    Apr 1   2018    Public
drwxr-xr-x  2   frank   frank   4096    Apr 1   2018    Templates
drwxr-xr-x  2   frank   frank   4096    Apr 1   2018    Videos
```

El primer caracter en la salida indica el tipo de archivo:

`-`
: para un archivo normal.

`d`
: para un directorio.

`c`
: para un archivo especial.

Para mostrar los tamaños de archivo en un formato legible para humanos, agregue la opción `-h`:

```
$ ls -lh
total 60K
drwxr-xr-x  2   frank   frank   4.0K    Apr 1   2018    Desktop
drwxr-xr-x  2   frank   frank   4.0K    Apr 1   2018    Documents
drwxr-xr-x  2   frank   frank   4.0K    Apr 1   2018    Downloads
-rw-r--r--  1   frank   frank     21    Sep 7   12:59   emp_name
-rw-r--r--  1   frank   frank     20    Sep 7   13:03   emp_salary
-rw-r--r--  1   frank   frank   8.8K    Apr 1   2018    examples.desktop
-rw-r--r--  1   frank   frank     10    Sep 1   2018    file1
-rw-r--r--  1   frank   frank     10    Sep 1   2018    file2
drwxr-xr-x  2   frank   frank   4.0K    Apr 1   2018    Music
drwxr-xr-x  2   frank   frank   4.0K    Apr 1   2018    Pictures
drwxr-xr-x  2   frank   frank   4.0K    Apr 1   2018    Public
drwxr-xr-x  2   frank   frank   4.0K    Apr 1   2018    Templates
drwxr-xr-x  2   frank   frank   4.0K    Apr 1   2018    Videos
```

Para enumerar todos los archivos, incluidos los archivos ocultos (los que comienzan con `.`) use la opción `-a`:

```
$ ls -a
.               .dbus   file1   .profile
..              Desktop file2   Public
.bash_history   .dmrc  .gconf  .sudo_as_admin_successful
```

Los archivos de configuración como `.bash_history` que están ocultos por defecto ahora están visibles.

En general, la sintaxis del comando `ls` viene dada por:

```
ls OPTIONS FILE
```

Donde `OPTIONS` son cualquiera de las opciones mostradas anteriormente (para ver todas las opciones posibles ejecute `man ls`), y `FILE` es el nombre del archivo o los detalles del directorio que desea listar.

NOTE Cuando no se especifica `FILE`, el directorio actual está implícito.

Crear, copiar, mover y eliminar archivos

Crear archivos con `touch`

El comando `touch` es la forma más fácil de crear archivos nuevos y vacíos. También puede usarlo para cambiar las marcas de tiempo (es decir, la hora de modificación) de los archivos y directorios existentes. La sintaxis para usar `touch` es:

```
touch OPTIONS FILE_NAME(S)
```

Sin ninguna opción, `touch` crearía nuevos archivos para cualquier nombre de archivo que se proporcione como argumento, siempre que los archivos con dichos nombres no existan. `touch` puede crear cualquier cantidad de archivos simultáneamente:

```
$ touch file1 file2 file3
```

Esto crearía tres nuevos archivos vacíos llamados `file1`, `file2` y `file3`.

Varias opciones de `touch` están específicamente diseñadas para permitir al usuario cambiar las marcas de tiempo de los archivos. Por ejemplo, la opción `-a` cambia solo la hora de acceso, mientras que la opción `-m` solo cambia la hora de modificación. El uso de ambas opciones juntas cambia el acceso y también la hora de modificación a la hora actual:

```
$ touch -am file3
```

Copiar archivos con `cp`

Como usuario de Linux, con frecuencia copiará archivos de una ubicación a otra. Ya sea que esté moviendo un archivo de música de un directorio a otro o un archivo del sistema, use `cp` para todas las tareas de copia:

```
$ cp file1 dir2
```

Este comando se puede interpretar literalmente como copiar `archivo1` en el directorio `dir2`. El resultado es la presencia de `file1` dentro de `dir2`. Para que este comando se ejecute correctamente, `file1` debe existir en el directorio actual del usuario. De lo contrario, el sistema informa un error con el mensaje `No such file or directory`.

```
$ cp dir1/file1 dir2
```

En este caso, observe que la ruta al `archivo1` es más explícita. La ruta de origen se puede expresar como *relativa* o *ruta absoluta*. Las rutas relativas se dan en referencia a un directorio específico, mientras que las rutas absolutas no se dan con una referencia. A continuación aclararemos más esta noción.

Por el momento, solo observe que este comando copia `file1` en el directorio `dir2`. La ruta a `file1` se proporciona con más detalle ya que el usuario no se encuentra actualmente en `dir1`.

```
$ cp /home/frank/Documents/file2 /home/frank/Documents/Backup
```

En este tercer caso, `file2` ubicado en `/home/frank/Documents` se copia en el directorio `/home/frank/Documents/Backup`. La ruta de origen proporcionada aquí es *absoluta*. En los dos ejemplos anteriores, las rutas de origen son *relativas*. Cuando una ruta comienza con el carácter `/` es una ruta absoluta; de lo contrario, es una ruta relativa.

La sintaxis general para `cp` es:

```
cp OPTIONS SOURCE DESTINATION
```

`SOURCE` es el archivo a copiar y `DESTINATION` el directorio en el que se copiaría el archivo. `SOURCE` y `DESTINATION` pueden especificarse como rutas absolutas o relativas.

Mover archivos con `mv`

Al igual que `cp` para copiar, Linux proporciona un comando para mover y renombrar archivos. Se llama `mv`.

La operación de mover es análoga a la operación de cortar y pegar que generalmente se realiza a través de una interfaz gráfica de usuario (GUI).

Si desea mover un archivo a una nueva ubicación, use `mv` de la siguiente manera:

```
mv FILENAME DESTINATION_DIRECTORY
```

Aquí hay un ejemplo:

```
$ mv myfile.txt /home/frank/Documents
```

El resultado es que `myfile.txt` se mueve al destino `/home/frank/Documents`.

Para renombrar un archivo, `mv` se usa de la siguiente manera:

```
$ mv old_file_name new_file_name
```

Esto cambia el nombre del archivo de `old_file_name` a `new_file_name`.

De manera predeterminada, `mv` no buscaría su confirmación (técnicamente dicho “would not prompt”) si desea sobrescribir (renombrar) un archivo existente. Sin embargo, puede hacer que el sistema solicite confirmación, utilizando la opción `-i`:

```
$ mv -i old_file_name new_file_name
mv: overwrite 'new_file_name'?
```

Este comando solicitaría el permiso del usuario antes de sobrescribir `old_file_name` a `new_file_name`.

Por el contrario, usando `-f`:

```
$ mv -f old_file_name new_file_name
```

sobrescribiría forzosamente el archivo, sin pedir ningún permiso.

Eliminar archivos con `rm`

`rm` se usa para eliminar archivos. Piense en ello como una forma abreviada de la palabra “remove”. Tenga en cuenta que la acción de eliminar un archivo suele ser irreversible, por lo que este comando debe utilizarse con precaución.

```
$ rm file1
```

Este comando eliminaría `file1`.

```
$ rm -i file1
rm: remove regular file 'file1'?
```

Este comando solicitaría la confirmación del usuario antes de eliminar `file1`. Recuerde, vimos la opción `-i` cuando usamos `mv` arriba.

```
$ rm -f file1
```

Este comando elimina forzosamente `file1` sin pedir su confirmación.

Se pueden eliminar varios archivos al mismo tiempo:

```
$ rm file1 file2 file3
```

En este ejemplo, `file1`, `file2` y `file3` se eliminan simultáneamente.

La sintaxis para `rm` generalmente viene dada por:

```
rm OPTIONS FILE
```

Crear y eliminar directorios

Crear directorios con `mkdir`

Crear directorios es fundamental para organizar sus archivos y carpetas. Los archivos se pueden agrupar de manera lógica manteniéndolos dentro de un directorio. Para crear un directorio, use `mkdir`:

```
mkdir OPTIONS DIRECTORY_NAME
```

donde `DIRECTORY_NAME` es el nombre del directorio que se creará. Se puede crear cualquier cantidad de directorios simultáneamente:

```
$ mkdir dir1
```

crearía el directorio `dir1` en el directorio actual del usuario.

```
$ mkdir dir1 dir2 dir3
```

El comando anterior crearía tres directorios `dir1`, `dir2` y `dir3` al mismo tiempo.

Para crear un directorio junto con sus subdirectorios, use la opción `-p` ("parents"):

```
$ mkdir -p parents/children
```

Este comando crearía la estructura de directorios `parents/children`, es decir, crearía los directorios `parents` y `children`. `children` se ubicarían dentro de `parents`.

Eliminar directorios con `rmdir`

`rmdir` borra un directorio *si está vacío*. Su sintaxis viene dada por:

```
rmdir OPTIONS DIRECTORY
```

donde `DIRECTORY` podría ser un argumento único o una lista de argumentos.

```
$ rmdir dir1
```

Este comando eliminaría `dir1`.

```
$ rmdir dir1 dir2
```

Este comando eliminaría simultáneamente `dir1` y `dir2`.

Puede eliminar un directorio con su subdirectorio:

```
$ rmdir -p parents/children
```

Esto eliminaría la estructura de directorios `parents/children`. Tenga en cuenta que si alguno de los directorios no está vacío, no se eliminarán.

Manipulación recursiva de archivos y directorios

Para manipular un directorio y su contenido, debe aplicar *recursión*. Recursión significa, hacer una acción y repetir esa acción en el árbol de directorios. En Linux, las opciones `-r` o `-R` o `--recursive` generalmente están asociadas con la recursividad.

El siguiente escenario lo ayudaría a comprender mejor la recursividad:

Usted lista el contenido de un directorio `students`, que contiene dos subdirectorios `level 1` y `level 2` y el archivo llamado `frank`. Al aplicar la recursividad, el comando `ls` enumeraría el contenido de `students`, es decir, `level 1`, `level 2` y `frank`, pero no terminaría allí. Ingresaría igualmente los subdirectorios `level 1` y `level 2` y listaría sus contenidos y así sucesivamente en el árbol de directorios.

Listado recursivo con `ls -R`

`ls -R` se usa para enumerar el contenido de un directorio junto con sus subdirectorios y archivos.

```
$ ls -R mydirectory
mydirectory/:
file1   newdirectory

mydirectory/newdirectory:
```

En el listado anterior, se muestra `mydirectory` incluyendo todo su contenido. Puede observar que `mydirectory` contiene el subdirectorio `newdirectory` y el archivo `file1`. `newdirectory` está vacío, por eso no se muestra contenido.

En general, para enumerar el contenido de un directorio, incluidos sus subdirectorios, use:

```
ls -R DIRECTORY_NAME
```

Agregar una barra inclinada final a `DIRECTORY_NAME` no tiene ningún efecto:

```
$ ls -R animal
```

es similar a

```
$ ls -R animal/
```

Copia recursiva con `cp -r`

`cp -r` (o `-R` o `--recursive`) le permite copiar un directorio junto con todos sus subdirectorios y archivos.

```
$ tree mydir
mydir
|_file1
|_newdir
    |_file2
    |_insidenew
        |_lastdir


3 directories, 2 files
$ mkdir newcopy
$ cp mydir newcopy
cp: omitting directory 'mydir'
$ cp -r mydir newcopy
* tree newcopy
newcopy
|_mydir
    |_file1
    |_newdir
        |_file2
        |_insidenew
            |_lastdir

4 directories, 2 files
```

En el listado anterior, observamos que al intentar copiar `mydir` en `newcopy`, usando `cp` sin `-r`, el sistema muestra el mensaje `cp: omitiendo el directorio 'mydir'`. Sin embargo, al agregar la opción `-r`, todo el contenido de `mydir`, incluido él mismo, se copia en `newcopy`.

Para copiar directorios y subdirectorios use:


```
cp -r SOURCE DESTINATION
```

Eliminación recursiva con `rm -r`

`rm -r` eliminará un directorio y todo su contenido (subdirectorios y archivos).

> **WARNING**
>
> Tenga mucho cuidado con la combinación `-r` o la opción de `-rf` cuando se usa con el comando `rm`. Un comando de eliminación recursivo en un directorio importante del sistema podría inutilizar el sistema. Emplee el comando de eliminación recursivo solo cuando esté absolutamente seguro de que el contenido de un directorio es seguro de eliminar de una computadora.

Al intentar eliminar un directorio sin usar `-r`, el sistema informaría un error:

```
$ rm newcopy/
rm: cannot remove 'newcopy/': Is a directory
$ rm -r newcopy/
```

Debe agregar `-r` como en el segundo comando para que la eliminación surta efecto.

> **NOTE**
>
> Quizás se pregunte por qué no usamos `rmdir` en este caso. Hay una sutil diferencia entre los dos comandos. `rmdir` lograría eliminar solo si el directorio dado está vacío, mientras que `rm -r` puede usarse independientemente de si este directorio está vacío o no.

Agregue la opción `-i` para pedir confirmación antes de eliminar el archivo:

```
$ rm -ri mydir/
rm: remove directory 'mydir/'?
```

El sistema solicita autorización antes de intentar eliminar `mydir`.

Archivos Globbing y Wildcards

El *globbing* es una característica proporcionada por el shell de Unix/Linux para representar múltiples nombres de archivo mediante el uso de caracteres especiales llamados *wildcards*. Los *wildcards* son esencialmente símbolos que pueden usarse para sustituir uno o más caracteres. Permiten, por ejemplo, mostrar todos los archivos que comienzan con la letra `A` o todos los archivos que terminan

con las letras `.conf`.

Los *wildcards* son muy útiles ya que pueden usarse con comandos como `cp`, `ls` o `rm`.

Los siguientes son algunos ejemplos de globbing:

`rm *`
: Elimina todos los archivos en el directorio de trabajo actual.

`ls l?st`
: Lista todos los archivos con nombres que comienzan con `l` seguidos de cualquier caracter individual y terminan con `st`.

`rmdir [a-z]*`
: Elimina todos los directorios cuyos nombres comienzan con una letra.

Tipos de Wildcards

Hay tres caracteres que se pueden usar como wildcards en Linux:

`*` (asterisco)
: que representa cero, una o más ocurrencias de cualquier carácter.

`?` (signo de interrogación)
: que representa una sola aparición de cualquier carácter.

`[]` (caracteres entre corchetes)
: que representa cualquier aparición de los caracteres encerrados entre corchetes. Es posible utilizar diferentes tipos de caracteres, ya sean números, letras u otros caracteres especiales. Por ejemplo, la expresión `[0-9]` coincide con todos los dígitos.

El Asterisco

Un asterisco (`*`) coincide con cero, una o más apariciones de cualquier carácter.

Por ejemplo:

```
$ find /home -name *.png
```

Esto encontraría todos los archivos que terminen con `.png` como `photo.png`, `cat.png`, `frank.png`. El

comando `find` se explorará más a fondo en la siguiente lección.

Similar:

```
$ ls lpic-*.txt
```

listaría todos los archivos de texto que comienzan con los caracteres `lpic-` seguidos de cualquier número de caracteres y terminan con `.txt`, como `lpic-1.txt` y `lpic-2.txt`.

El wildcard asterisco se puede utilizar para manipular (copiar, eliminar o mover) todo el contenido de un directorio:

```
$ cp -r animal/* forest
```

En este ejemplo, todos los contenidos de `animal` se copian en `forest`.

En general, para copiar todos los contenidos de un directorio, utilizamos:

```
cp -r SOURCE_PATH/* DEST_PATH
```

donde `SOURCE_PATH` se puede omitir si ya estamos en el directorio requerido.

El asterisco, como cualquier otro *wildcard*, podría usarse repetidamente en el mismo comando y en cualquier posición:

```
$ rm *ate*
```

Los nombres de archivo con el prefijo cero, una o más apariciones de cualquier carácter, seguidos de las letras `ate` y terminando con cero, una o más ocurrencias se eliminarán.

El signo de interrogación

El signo de interrogación (`?`) Coincide con una *única* ocurrencia de un caracter.

Considere el listado:

```
$ ls
last.txt    lest.txt    list.txt    third.txt   past.txt
```

Para devolver solo los archivos que comienzan con `l` seguidos de cualquier caracter individual y los caracteres `st.txt`, utilizamos el comodín de signo de interrogación (?):

```
$ ls l?st.txt
last.txt    lest.txt    list.txt
```

Solo se muestran los archivos `last.txt`, `lest.txt` y `list.txt`, ya que coinciden con los criterios dados.

Similar,

```
$ ls ??st.txt
last.txt    lest.txt    list.txt    past.txt
```

archivos de salida con el prefijo de dos caracteres seguidos del texto `st.txt`.

Caracteres entre Corchetes

Los *wildcard* entre corchetes coinciden con cualquier aparición de los caracteres entre corchetes:

```
$ ls l[aef]st.txt
last.txt    lest.txt
```

Este comando listaría todos los archivos que comienzan con `l` seguido de *cualquiera* de los caracteres en el conjunto `aef` y terminan con `st.txt`.

Los corchetes también podrían tomar rangos:

```
$ ls l[a-z]st.txt
last.txt    lest.txt    list.txt
```

Esto genera todos los archivos con nombres que comienzan con `l` seguidos de cualquier letra minúscula en el rango de `a` a `z` y terminan con `st.txt`.

También se pueden aplicar varios rangos entre corchetes:

```
$ ls
student-1A.txt  student-2A.txt  student-3.txt
$ ls student-[0-9][A-Z].txt
student-1A.text student-2A.txt
```

El listado muestra un directorio escolar con una lista de estudiantes registrados. Para enumerar solo aquellos estudiantes cuyos números de registro cumplen los siguientes criterios:

- comienza con `student-`
- seguido de un número y un carácter en mayúscula
- y termina con `.txt`

Combinando Wildcards

Los *wildcards* se pueden combinar de la siguiente forma:

```
$ ls
last.txt    lest.txt    list.txt    third.txt   past.txt
$ ls [plf]?st*
last.txt    lest.txt    list.txt    past.txt
```

El primer componente *wildcard* (`[plf]`) coincide con cualquiera de los caracteres `p`, `l` o `f`. El segundo componente *wildcard* (`?`) coincide con cualquier carácter individual. El tercer componente *wildcard* (`*`) coincide con cero, una o varias ocurrencias de cualquier caracter.

```
$ ls
file1.txt file.txt file23.txt fom23.txt
$ ls f*[0-9].txt
file1.txt file23.txt fom23.txt
```

El comando anterior muestra todos los archivos que comienzan con la letra `f`, seguido de cualquier conjunto de letras, al menos una aparición de un dígito y termina con `.txt`. Tenga en cuenta que `file.txt` no se muestra ya que no coincide con este criterio.

Ejercicios Guiados

1. Considere la siguiente lista:

```
$ ls -lh
total 60K
drwxr-xr-x  2   frank   frank   4.0K    Apr 1   2018    Desktop
drwxr-xr-x  2   frank   frank   4.0K    Apr 1   2018    Documents
drwxr-xr-x  2   frank   frank   4.0K    Apr 1   2018    Downloads
-rw-r--r--  1   frank   frank     21    Sep 7   12:59   emp_name
-rw-r--r--  1   frank   frank     20    Sep 7   13:03   emp_salary
-rw-r--r--  1   frank   frank   8.8K    Apr 1   2018    examples.desktop
-rw-r--r--  1   frank   frank     10    Sep 1   2018    file1
-rw-r--r--  1   frank   frank     10    Sep 1   2018    file2
drwxr-xr-x  2   frank   frank   4.0K    Apr 1   2018    Music
drwxr-xr-x  2   frank   frank   4.0K    Apr 1   2018    Pictures
drwxr-xr-x  2   frank   frank   4.0K    Apr 1   2018    Public
drwxr-xr-x  2   frank   frank   4.0K    Apr 1   2018    Templates
drwxr-xr-x  2   frank   frank   4.0K    Apr 1   2018    Videos
```

   - ¿Qué representa el caracter `d` en la salida?

   - ¿Por qué los tamaños se dan en formato legible para humanos?

   - ¿Cuál sería la diferencia en la salida si se usara `ls` sin argumentos?

2. Considere el siguiente comando:

```
$ cp /home/frank/emp_name /home/frank/backup
```

   - ¿Qué pasaría con el archivo `emp_name` si este comando se ejecuta con éxito?

   - Si `emp_name` era un directorio, ¿qué opción debería agregarse a `cp` para ejecutar el comando?

- Si `cp` ahora se cambia a `mv`, ¿qué resultados espera?

3. Considere el listado:

```
$ ls
file1.txt file2.txt file3.txt file4.txt
```

¿Qué *wildcard* ayudaría a eliminar todo el contenido de este directorio?

4. Según el listado anterior, ¿qué archivos se mostrarían con el siguiente comando?

```
$ ls file*.txt
```

5. Complete el comando agregando los dígitos y caracteres apropiados entre corchetes que listarían todo el contenido anterior:

```
$ ls file[].txt
```

Ejercicios Exploratorios

1. En su directorio de inicio, cree los archivos llamados `dog` y `cat`.
2. Aún en su directorio de inicio, cree el directorio llamado `animal`. Mueva `dog` y `cat` a `animal`.
3. Vaya a la carpeta `Documents` que se encuentra en su directorio de inicio y dentro, cree el directorio `backup`.
4. Copie `animal` y su contenido en `backup`.
5. Cambie el nombre de `animal` en `backup` a `animal.bkup`.
6. El directorio `/home/lpi/bases de datos` contiene varios archivos incluyendo: `db-1.tar.gz`, `db-2.tar.gz` y `db-3.tar.gz`. ¿Qué comando único puede usar para listar solo los archivos mencionados anteriormente?

7. Considere el listado:

```
$ ls
cne1222223.pdf cne12349.txt cne1234.pdf
```

Con el uso de un solo carácter globbing, ¿qué comando eliminaría solo los archivos pdf?

Resumen

En esta lección, exploramos cómo ver qué hay dentro de un directorio con el comando `ls`, cómo copiar archivos y carpetas (`cp`) y cómo moverlos (`mv`). También observamos cómo se pueden crear nuevos directorios con el comando `mkdir`. También se discutieron los comandos para eliminar archivos (`rm`) y carpetas (`rmdir`).

En esta lección, también aprendió sobre *globbing* y *wildcards*. El *globbing* se utiliza para representar múltiples nombres de archivos mediante el uso de caracteres especiales llamados *wildcards*. Los *wildcards* básicos y sus significados:

`?` (signo de interrogación)
: representa una sola aparición de un caracter.

`[]` (corchetes)
: representa cualquier aparición de los caracteres encerrados entre corchetes.

`*` (asterisco)
: representa cero, una o más ocurrencias de cualquier caracter.

Puede combinar cualquiera de estos *wildcards* en la misma declaración.

Respuestas a los ejercicios guiados

1. Considere la siguiente lista:

```
$ ls -lh
total 60K
drwxr-xr-x  2   frank   frank   4.0K    Apr 1   2018    Desktop
drwxr-xr-x  2   frank   frank   4.0K    Apr 1   2018    Documents
drwxr-xr-x  2   frank   frank   4.0K    Apr 1   2018    Downloads
-rw-r--r--  1   frank   frank     21    Sep 7   12:59   emp_name
-rw-r--r--  1   frank   frank     20    Sep 7   13:03   emp_salary
-rw-r--r--  1   frank   frank   8.8K    Apr 1   2018    examples.desktop
-rw-r--r--  1   frank   frank     10    Sep 1   2018    file1
-rw-r--r--  1   frank   frank     10    Sep 1   2018    file2
drwxr-xr-x  2   frank   frank   4.0K    Apr 1   2018    Music
drwxr-xr-x  2   frank   frank   4.0K    Apr 1   2018    Pictures
drwxr-xr-x  2   frank   frank   4.0K    Apr 1   2018    Public
drwxr-xr-x  2   frank   frank   4.0K    Apr 1   2018    Templates
drwxr-xr-x  2   frank   frank   4.0K    Apr 1   2018    Videos
```

   - ¿Qué representa el caracter `d` en la salida?

     `d` es el caracter que identifica un directorio.

   - ¿Por qué los tamaños se dan en formato legible para humanos?

     Debido a la opción `-h`.

   - ¿Cuál sería la diferencia en la salida si se usara `ls` sin argumentos?

     Solo se proporcionarán los nombres de directorios y archivos.

2. Considere el siguiente comando:

```
$ cp /home/frank/emp_name /home/frank/backup
```

   - ¿Qué pasaría con el archivo `emp_name` si este comando se ejecuta con éxito?

     `emp_name` se copiaría en `backup`.

   - Si `emp_name` era un directorio, ¿qué opción debería agregarse a `cp` para ejecutar el comando?

`-r`

- Si `cp` ahora se cambia a `mv`, ¿qué resultados espera?

  `emp_name` se movería a `backup`. Ya no estaría presente dentro del directorio de usuario `frank`.

3. Considere el listado:

```
$ ls
file1.txt file2.txt file3.txt file4.txt
```

¿Qué *wildcard* ayudaría a eliminar todo el contenido de este directorio?

El asterisco `*`.

4. Según el listado anterior, ¿qué archivos se mostrarían con el siguiente comando?

```
$ ls file*.txt
```

Todos ellos, ya que el asterisco representa cualquier número de caracteres.

5. Complete el comando agregando los dígitos y caracteres apropiados entre corchetes que listarían todo el contenido anterior:

```
$ ls file[].txt
```

`file[0-9].txt`

Respuestas a ejercicios exploratorios

1. En su directorio de inicio, cree los archivos llamados `dog` y `cat`.

```
$ touch dog cat
```

2. Aún en su directorio de inicio, cree el directorio llamado `animal`. Mueva `dog` y `cat` a `animal`.

```
$ mkdir animal
$ mv dog cat -t animal/
```

3. Vaya a la carpeta `Documents` que se encuentra en su directorio de inicio y dentro, cree el directorio `backup`.

```
$ cd ~/Documents
$ mkdir backup
```

4. Copie `animal` y su contenido en `backup`.

```
$ cp -r animal ~/Documents/backup
```

5. Cambie el nombre de `animal` en `backup` a `animal.bkup`.

```
$ mv animal/ animal.bkup
```

6. El directorio `/home/lpi/bases de datos` contiene varios archivos incluyendo: `db-1.tar.gz`, `db-2.tar.gz` y `db-3.tar.gz`. ¿Qué comando único puede usar para listar solo los archivos mencionados anteriormente?

```
$ ls db-[1-3].tar.gz
```

7. Considere el listado:

```
$ ls
cne1222223.pdf cne12349.txt cne1234.pdf
```

Con el uso de un solo carácter globbing, ¿qué comando eliminaría solo los archivos pdf?

```
$ rm *.pdf
```



103.3 Lección 2

Certificación:	LPIC-1
Versión:	5.0
Tema:	103 Comandos GNU y Unix
Objetivo:	103.3 Gestión básica de archivos
Lección:	2 de 2

Introducción

Cómo encontrar archivos

A medida que utiliza su máquina, los archivos crecen progresivamente en número y tamaño. A veces se hace difícil localizar un archivo en particular. Afortunadamente, Linux proporciona `find` para buscar y localizar archivos rápidamente. `find` usa la siguiente sintaxis:

```
find STARTING_PATH OPTIONS EXPRESSION
```

`STARTING_PATH`
: define el directorio donde comienza la búsqueda.

`OPTIONS`
: controla el comportamiento y agrega criterios específicos para optimizar el proceso de búsqueda.

`EXPRESSION`
: define la consulta de búsqueda.

```
$ find . -name "myfile.txt"
./myfile.txt
```

La ruta de inicio en este caso es el directorio actual. La opción `-name` especifica que la búsqueda se basa en el nombre del archivo. `myfile.txt` es el nombre del archivo a buscar. Cuando utilice *globbing*, asegúrese de incluir la expresión entre comillas:

```
$ find /home/frank -name "*.png"
/home/frank/Pictures/logo.png
/home/frank/screenshot.png
```

Este comando encuentra todos los archivos que terminan con `.png` comenzando desde el directorio `/home/frank/` y dentro de éste. Si no comprende el uso del asterisco (*), se trata en la lección anterior.

Uso de criterios para acelerar la búsqueda

Use `find` para localizar archivos basados en *tipo*, *tamaño* u *hora*. Al especificar una o más opciones, los resultados deseados se obtienen en menos tiempo.

Los cambios para buscar archivos según el tipo incluyen:

`-type f`
búsqueda de archivos.

`-type d`
búsqueda de directorio.

`-type l`
búsqueda de enlaces simbólicos.

```
$ find . -type d -name "example"
```

Este comando encuentra todos los directorios en el directorio actual y dentro de éste, que tienen el nombre `example`.

Otros criterios que podrían usarse con `find` incluyen:

`-name`

realiza una búsqueda basada en el nombre dado.

`-iname`

búsquedas basadas en el nombre, sin embargo, las mayúsculas y minúsculas no son importantes (es decir, el caso de prueba `myFile` es similar a `MYFILE`).

`-not`

devuelve los resultados que *no* coinciden con el caso de prueba.

`-maxdepth N`

busca en el directorio actual así como en los subdirectorios `N` niveles de profundidad.

Localización de archivos por hora de modificación

`find` también permite filtrar una jerarquía de directorios en función de cuándo se modificó el archivo:

```
$ sudo find / -name "*.conf" -mtime 7
/etc/logrotate.conf
```

Este comando buscaría todos los archivos en todo el sistema de archivos (la ruta de inicio es el directorio raíz, es decir, `/`) que termina con los caracteres `.conf` y se ha modificado en los últimos siete días. Este comando requeriría privilegios elevados para acceder a los directorios comenzando en la base de la estructura de directorios del sistema, de ahí el uso de `sudo` aquí. El argumento pasado a `mtime` representa el *número de días* desde la última modificación del archivo.

Ubicar archivos por tamaño

`find` también puede localizar archivos por *tamaño*. Por ejemplo, buscando archivos más grandes que `2G` en `/var`:

```
$ sudo find /var -size +2G
/var/lib/libvirt/images/debian10.qcow2
/var/lib/libvirt/images/rhel8.qcow2
```

La opción `-size` muestra archivos de tamaños correspondientes al argumento pasado. Algunos argumentos de ejemplo incluyen:

`-size 100b`
: archivos de exactamente 100 bytes.

`-size +100k`
: archivos de más de 100 kilobytes.

`-size -20M`
: archivos de menos de 20 megabytes.

`-size +2G`
: archivos de más de 2 gigabytes.

> **NOTE** Para encontrar archivos vacíos podemos usar: `find . -size 0b` o `find . -empty`.

Actuando sobre el conjunto de resultados

Una vez que se realiza una búsqueda, es posible realizar una acción en el conjunto resultante utilizando `-exec`:

```
$ find . -name "*.conf" -exec chmod 644 '{}' \;
```

Esto filtra cada objeto en el directorio actual (`.`) y dentro de éste para los nombres de archivo que terminan en `x.conf` y luego ejecuta el comando `chmod 644` para modificar los permisos de archivo en los resultados.

Por ahora, no se preocupe con el significado de `'{}'\;` ya que se discutirá más adelante.

Usar `grep` para filtrar archivos basados en contenido

`grep` se usa para buscar la aparición de una palabra clave.

Considere una situación en la que debemos encontrar archivos basados en el contenido:

```
$ find . -type f -exec grep "lpi" '{}' \; -print
./.bash_history
Alpine/M
helping/M
```

Esto buscaría cada objeto en la jerarquía de directorio actual (`.`) en archivos (`-type f`) y luego

ejecuta el comando `grep "lpi"` para cada archivo que satisfaga las condiciones. Los archivos que coinciden con estas condiciones se imprimen en la pantalla (`-print`). Las llaves (`{}`) son un marcador de posición para los resultados de la búsqueda `find`. Los `{}` están encerrados entre comillas simples (`'`) para evitar pasar archivos `grep` con nombres que contengan caracteres especiales. El comando `-exec` se termina con un punto y coma (`;`), que se debe escapar (`\;`) para evitar la interpretación por parte del shell.

Agregar la opción `-delete` al final de una expresión eliminaría todos los archivos que coincidan. Esta opción debe usarse cuando esté seguro de que los resultados solo coinciden con los archivos que desea eliminar.

En el ejemplo a continuación, `find` localiza todos los archivos en la jerarquía comenzando en el directorio actual y luego elimina todos los archivos que terminan con los caracteres `.bak`:

```
$ find . -name "*.bak" -delete
```

Archivado de archivos

El comando `tar` (archivo y compresión)

El comando `tar`, abreviatura de "tape archive (r)", se utiliza para crear archivos tar convirtiendo un grupo de archivos en un archivo. Los archivos se crean para mover o respaldar fácilmente un grupo de archivos. Piense en `tar` como una herramienta que crea un pegamento sobre el cual los archivos se pueden adjuntar, agrupar y mover fácilmente.

`tar` también tiene la capacidad de extraer archivos tar, mostrar una lista de los archivos incluidos en el archivo y agregar archivos adicionales a un archivo existente.

La sintaxis del comando `tar` es la siguiente:

```
tar [OPERATION_AND_OPTIONS] [ARCHIVE_NAME] [FILE_NAME(S)]
```

`OPERATION`

Solo se permite y requiere un argumento de operación. Las operaciones más utilizadas son:

`--create` (`-c`)

Crea un nuevo archivo tar.

`--extract` (`-x`)
: Extrae todo el archivo o uno o más archivos de un archivo.

`--list` (`-t`)
: Muestra una lista de los archivos incluidos en el archivo.

`OPTIONS`
: Las opciones más utilizadas son:

`--verbose` (`-v`)
: Muestra los archivos que se están procesando con el comando `tar`.

`--file=archive-name` (`-f archive-name`)
: Especifica el nombre del archivo de almacenamiento.

`ARCHIVE_NAME`
: El nombre del archivo.

`FILE_NAME (S)`
: Una lista de nombres de archivos separados por espacios que se extraerán. Si no se proporciona, se extrae todo el archivo.

Crear un archivo

Supongamos que tenemos un directorio llamado `stuff` en el directorio actual y queremos guardarlo en un archivo llamado `archive.tar`. Ejecutaríamos el siguiente comando:

```
$ tar -cvf archive.tar stuff
stuff/
stuff/service.conf
```

Esto es lo que realmente significan esos parámetros:

`-c`
: Crea un archivo.

`-v`
: Muestra el progreso en la terminal mientras se crea el archivo, también conocido como modo "verbose". `-v` siempre es opcional en estos comandos, pero es útil.

-f
  Permite especificar el nombre de archivo del archivo.

En general, para archivar un único directorio o un único archivo en Linux, utilizamos:

```
tar -cvf NAME-OF-ARCHIVE.tar /PATH/TO/DIRECTORY-OR-FILE
```

> **NOTE** `tar` funciona de forma recursiva. Realizará la acción requerida en cada directorio posterior dentro del directorio especificado.

Para archivar múltiples directorios a la vez, se especifican todos los directorios delimitándolos por un espacio en la sección `/PATH/TO/DIRECTORY-OR-FILE`:

```
$ tar -cvf archive.tar stuff1 stuff2
```

Esto produciría un archivo de `stuff1` y `stuff2` en `archive.tar`

Extraer un archivo

Podemos extraer un archivo usando `tar`:

```
$ tar -xvf archive.tar
stuff/
stuff/service.conf
```

Esto extraerá el contenido de `archive.tar` en el directorio actual.

Este comando es el mismo que el comando de creación de archivo utilizado anteriormente, excepto el modificador `-x` que reemplaza al modificador `-c`.

Para extraer el contenido del archivo en un directorio específico, usamos `-C`:

```
$ tar -xvf archive.tar -C /tmp
```

Esto extraerá el contenido de `archive.tar` al directorio `/tmp`.

```
$ ls /tmp
stuff
```

Comprimiendo con `tar`

El comando GNU `tar` incluido en las distribuciones de Linux puede crear un archivo `.tar` y luego comprimirlo con la compresión `gzip` o `bzip2` en un solo comando:

```
$ tar -czvf name-of-archive.tar.gz stuff
```

Este comando crearía un archivo comprimido usando el algoritmo `gzip` (`-z`).

Si bien la compresión `gzip` se usa con mayor frecuencia para crear archivos `.tar.gz` o `.tgz`, `tar` también admite la compresión `bzip2`. Esto permite la creación de archivos comprimidos `bzip2`, a menudo llamados archivos `.tar.bz2`, `.tar.bz` o `.tbz`.

Para hacerlo, reemplazamos `-z` por `gzip` con `-j` por `bzip2`:

```
$ tar -cjvf name-of-archive.tar.bz stuff
```

Para descomprimir el archivo, reemplazamos `-c` por `-x`, donde `x` significa "extract":

```
$ tar -xzvf archive.tar.gz
```

`gzip` es más rápido, pero generalmente se comprime un poco menos, por lo que se obtiene un archivo algo más grande. `bzip2` es más lento, pero se comprime un poco más, por lo que se obtiene un archivo algo más pequeño. Sin embargo, en general, `gzip` y `bzip2` son prácticamente lo mismo y ambos funcionarán de manera similar.

Alternativamente, podemos aplicar la compresión `gzip` o `bzip2` usando el comando `gzip` para las compresiones `gzip` y el comando `bzip` para las compresiones `bzip`. Por ejemplo, para aplicar la compresión `gzip`, use:

```
gzip FILE-TO-COMPRESS
```

`gzip`
: crea el archivo comprimido con el mismo nombre pero con un final `.gz`.

`gzip`
: elimina los archivos originales después de crear el archivo comprimido.

El comando `bzip2` funciona de manera similar.

Para descomprimir los archivos, usamos `gunzip` o `bunzip2` dependiendo del algoritmo utilizado para comprimir un archivo.

El comando `cpio`

`cpio` significa "copy in, copy out" (*copiar dentro, copiar fuera*). Se utiliza para procesar archivos de almacenamiento como archivos `*.cpio` o `*.tar`.

`cpio` realiza las siguientes operaciones:

- Copiar archivos a un archivo.
- Extraer archivos de un archivo.

Toma la lista de archivos de la entrada estándar (principalmente salida de `ls`).

Para crear un archivo `cpio`, utilizamos:

```
$ ls | cpio -o > archive.cpio
```

La opción `-o` indica a `cpio` que cree una salida. En este caso, el archivo de salida creado es `archive.cpio`. El comando `ls` lista el contenido del directorio actual que se archivará.

Para extraer el archivo utilizamos:

```
$ cpio -id < archive.cpio
```

La opción `-i` se usa para realizar el extracto. La opción `-d` crearía la carpeta de destino. El caracter `<` representa la entrada estándar. El archivo de entrada a extraer es `archive.cpio`.

El comando `dd`

`dd` copia datos de una ubicación a otra. La sintaxis de la línea de comandos de `dd` difiere de muchos otros programas de Unix, utiliza la sintaxis `option=value` para sus opciones de línea de comandos en lugar de los formatos estándar GNU `-option value` o `--option=value`:

```
$ dd if=oldfile of=newfile
```

Este comando copiará el contenido de `oldfile` en `newfile`, donde `if=` es el archivo de entrada y `of=` se refiere al archivo de salida.

> **NOTE** El comando `dd` normalmente no mostrará nada en la pantalla hasta que el comando haya finalizado. Al proporcionar la opción `status=progress`, la consola mostrará la cantidad de trabajo que realiza el comando. Por ejemplo: `dd status=progress if=oldfile of=newfile`.

`dd` también se utiliza para cambiar datos a mayúsculas/minúsculas o escribir directamente en dispositivos de bloque como `/dev/sdb`:

```
$ dd if=oldfile of=newfile conv=ucase
```

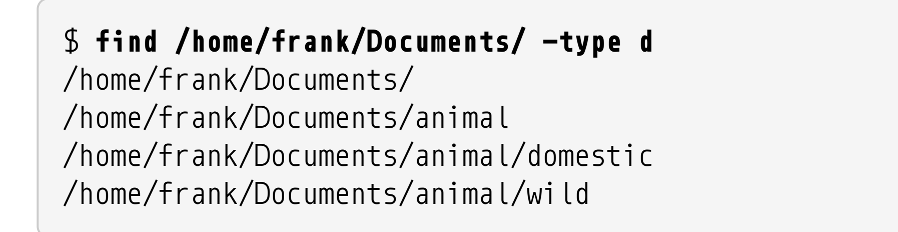
Esto copiaría todo el contenido de `oldfile` en `newfile` y capitalizaría todo el texto.

El siguiente comando hará una copia de seguridad de todo el disco duro ubicado en `/dev/sda` en un archivo llamado `backup.dd`:

```
$ dd if=/dev/sda of=backup.dd bs=4096
```

Ejercicios Guiados

1. Considere la siguiente lista:



```
$ find /home/frank/Documents/ -type d
/home/frank/Documents/
/home/frank/Documents/animal
/home/frank/Documents/animal/domestic
/home/frank/Documents/animal/wild
```

   - ¿Qué tipo de archivos generaría este comando?

   - ¿En qué directorio comienza la búsqueda?

2. Un usuario desea comprimir su carpeta de respaldo. Él usa el siguiente comando:

```
$ tar cvf /home/frank/backup.tar.gz /home/frank/dir1
```

¿Qué opción falta para comprimir la copia de seguridad utilizando el algoritmo `gzip`?

Ejercicios Exploratorios

1. Como administrador del sistema, es necesario realizar verificaciones regulares para eliminar archivos voluminosos. Estos voluminosos archivos se encuentran en `/var` y terminan con una extensión `.backup`.

   - Escriba el comando, usando `find`, para localizar estos archivos:

   - Un análisis de los tamaños de estos archivos revela que varían de `100M` a `1G`. Complete el comando anterior con esta nueva información, para que pueda ubicar esos archivos de respaldo que van desde `100M` a `1G`:

   - Finalmente, complete este comando, con la acción de eliminación para que se eliminen estos archivos:

2. En el directorio `/var`, existen cuatro archivos de respaldo:

```
db-jan-2018.backup
db-feb-2018.backup
db-march-2018.backup
db-apr-2018.backup
```

   - Usando `tar`, especifique el comando que crearía un archivo con el nombre `db-first-quarter-2018.backup.tar`:

   - Usando `tar`, especifique el comando que crearía el archivo comprimido y comprímalo usando `gzip`. Tenga en cuenta que el nombre del archivo resultante debe terminar con `.gz`:

Resumen

En esta sección, aprendió:

- ¿Cómo encontrar archivos con `find`?
- ¿Cómo agregar criterios de búsqueda basados en el tiempo, tipo de archivo o tamaño al proporcionar un argumento para `find`?
- ¿Cómo actuar en un conjunto devuelto?
- ¿Cómo archivar, comprimir y descomprimir archivos usando `tar`?
- Procesar archivos con `cpio`.
- Copiar archivos con `dd`.

Respuestas a los ejercicios guiados

1. Considere la siguiente lista:

```
$ find /home/frank/Documents/ -type d
/home/frank/Documents/
/home/frank/Documents/animal
/home/frank/Documents/animal/domestic
/home/frank/Documents/animal/wild
```

   - ¿Qué tipo de archivos generaría este comando?

     Directorios.

   - ¿En qué directorio comienza la búsqueda?

     `/home/frank/Documents`

2. Un usuario desea comprimir su carpeta de respaldo. Él usa el siguiente comando:

```
$ tar cvf /home/frank/backup.tar.gz /home/frank/dir1
```

   ¿Qué opción falta para comprimir la copia de seguridad utilizando el algoritmo `gzip`?

   Option `-z`.

Respuestas a ejercicios exploratorios

1. Como administrador del sistema, es necesario realizar verificaciones regulares para eliminar archivos voluminosos. Estos voluminosos archivos se encuentran en `/var` y terminan con una extensión `.backup`.
   - Escriba el comando, usando `find`, para localizar estos archivos:

     ```
     $ find /var -name *.backup
     ```

   - Un análisis de los tamaños de estos archivos revela que varían de `100M` a `1000M`. Complete el comando anterior con esta nueva información, para que pueda ubicar esos archivos de respaldo que van desde `100M` a `1000M`:

     ```
     $ find /var -name *.backup -size +100M -size -1000M
     ```

   - Finalmente, complete este comando, con la acción de eliminación para que se eliminen estos archivos:

     ```
     $ find /var -name *.backup -size +100M -size -1000M -delete
     ```

2. En el directorio `/var`, existen cuatro archivos de respaldo:

   ```
   db-jan-2018.backup
   db-feb-2018.backup
   db-march-2018.backup
   db-apr-2018.backup
   ```

   - Usando `tar`, especifique el comando que crearía un archivo con el nombre `db-first-quarter-2018.backup.tar`:

     ```
     $ tar -cvf db-first-quarter-2018.backup.tar db-jan-2018.backup db-feb-
     2018.backup db-march-2018.backup db-apr-2018.backup
     ```

   - Usando `tar`, especifique el comando que crearía el archivo comprimido y comprímalo usando `gzip`. Tenga en cuenta que el nombre del archivo resultante debe terminar con `.gz`:


```
$ tar -zcvf db-first-quarter-2018.backup.tar.gz db-jan-2018.backup db-feb-
2018.backup db-march-2018.backup db-apr-2018.backup
```



103.4 Uso de secuencias de texto, tuberías y redireccionamientos

Referencia al objetivo del LPI

LPIC-1 v5, Exam 101, Objective 103.4

Importancia

4

Áreas de conocimiento clave

- Redireccionar la entrada estándar (stdin), la salida estándar (stdout) y el error estándar (stderr).
- Utilizar tuberías para enviar la salida de un comando a la entrada de otro.
- Usar la salida de un comando como argumento de otro comando.
- Enviar la salida de un comando a stdouty a un archivo simultáneamente.

Lista parcial de archivos, términos y utilidades

- `tee`
- `xargs`



103.4 Lección 1

Certificación:	LPIC-1 (101)
Versión:	5.0
Tema:	103 Comandos GNU y Unix
Objetivo:	103.4 Usar flujos, tuberías y redireccionamientos
Lección:	1 de 2

Introducción

Todos los programas de computadora siguen el mismo principio general: los datos recibidos de alguna fuente se transforman para generar un resultado inteligible. En el contexto de shell de Linux, la fuente de datos puede ser un archivo local, un archivo remoto, un dispositivo (como un teclado), etc. La salida del programa generalmente se representa en una pantalla, pero también es común almacenar los datos de salida en un sistema de archivos local, enviarlo a un dispositivo remoto, reproducirlo a través de altavoces de audio, etc.

Los sistemas operativos inspirados en Unix, como Linux, ofrecen una gran variedad de métodos de entrada/salida. En particular, el método de descriptores de archivos permite asociar dinámicamente números enteros con canales de datos, de modo que un proceso pueda hacer referencia a ellos como sus flujos de datos de entrada/salida.

Los procesos estándar de Linux tienen tres canales de comunicación abiertos de manera predeterminada: el canal de entrada estándar (la mayoría de las veces llamado simplemente *stdin*), el *canal de salida* estándar (*stdout*) y el *canal de error* estándar (*stderr*). Los descriptores numéricos de archivos asignados a estos canales son `0` a stdin, `1` a stdout y `2` a stderr. Los canales de comunicación también son accesibles a través de los dispositivos especiales `/dev/stdin`, `/dev/stdout` y

`/dev/stderr`.

Estos tres canales de comunicación estándar permiten a los programadores escribir código que lee y escribe datos sin preocuparse por el tipo de medio del que proviene o al que va. Por ejemplo, si un programa necesita un conjunto de datos como entrada, solo puede solicitar datos de la entrada estándar y lo que se esté utilizando como entrada estándar proporcionará esos datos. Del mismo modo, el método más simple que un programa puede usar para mostrar su salida es escribirlo en la salida estándar. En una sesión de shell estándar, el teclado se define como stdin y la pantalla del monitor se define como stdout y stderr.

El shell Bash tiene la capacidad de reasignar los canales de comunicación al cargar un programa. Permite, por ejemplo, anular la pantalla como salida estándar y usar un archivo en el sistema de archivos local como stdout.

Redireccionamientos

La reasignación del descriptor de archivo de un canal en el entorno de shell se denomina *redirect*. Una redirección se define mediante un caracter especial dentro de la línea de comandos. Por ejemplo, para redirigir la salida estándar de un proceso a un archivo, el símbolo *mayor que* `>` se coloca al final del comando y sigue la ruta al archivo que recibirá la salida redirigida:

```
$ cat /proc/cpuinfo >/tmp/cpu.txt
```

Por defecto, solo se redirige el contenido que llega a stdout. Eso sucede porque el valor numérico del descriptor de archivo debe especificarse justo antes del símbolo mayor que y, cuando no se especifica, Bash redirige la salida estándar. Por lo tanto, usar `>` es equivalente a usar `1>` (el valor del descriptor de archivo stdout es `1`).

Para capturar el contenido de stderr, se debe usar la redirección `2>` en su lugar. La mayoría de los programas de línea de comandos envían información de depuración y mensajes de error al canal de error estándar. Es posible, por ejemplo, capturar el mensaje de error provocado por un intento de leer un archivo inexistente:

```
$ cat /proc/cpu_info 2>/tmp/error.txt
$ cat /tmp/error.txt
cat: /proc/cpu_info: No such file or directory
```

Tanto stdout como stderr se redirigen al mismo objetivo con `&>` o `>&`. Es importante no colocar ningún espacio al lado del ampersand, de lo contrario, Bash lo tomará como la instrucción para ejecutar el proceso en segundo plano y no realizar la redirección.

El destino debe ser una ruta a un archivo en el que se pueda escribir, como `/tmp/cpu.txt`, o un descriptor de archivo editable. Un objetivo de descriptor de archivo está representado por un ampersand seguido del valor numérico del descriptor de archivo. Por ejemplo, `1>&2` redirige stdout a stderr. Para hacer lo contrario, de stderr a stdout, se debe usar `2>&1` en su lugar.

Aunque no es muy útil, dado que hay una forma más corta de hacer la misma tarea, es posible redirigir stderr a stdout y luego redirigirlo a un archivo. Por ejemplo, una redirección para escribir stderr y stdout en un archivo llamado `log.txt` puede escribirse como `>log.txt 2>&1`. Sin embargo, la razón principal para redirigir stderr a stdout es permitir el análisis de mensajes de error y depuración. Es posible redirigir la salida estándar de un programa a la entrada estándar de otro, pero no es posible redirigir directamente el error estándar a la entrada estándar de otro programa. Por lo tanto, los mensajes del programa enviados a stderr primero deben redirigirse a stdout para que otros stdin puedan leerlos.

Para descartar la salida de un comando, su contenido se puede redirigir al archivo especial `/dev/null`. Por ejemplo, `>log.txt 2>/dev/null` guarda el contenido de stdout en el archivo `log.txt` y descarta el stderr. El archivo `/dev/null` puede ser escrito por cualquier usuario pero no se pueden recuperar datos, ya que no se almacenan en ningún lado.

Se presenta un mensaje de error si el destino especificado no se puede escribir (si la ruta apunta a un directorio o un archivo de solo lectura) y no se realiza ninguna modificación en el destino. Sin embargo, una redirección de salida sobrescribe el destino existente sin ninguna confirmación. Los archivos se sobrescriben mediante redireccionamientos de salida a menos que la opción Bash `noclobber` esté habilitada, lo que se puede hacer para la sesión actual con el comando `set -o noclobber` o `set -C`:

```
$ set -o noclobber
$ cat /proc/cpu_info 2>/tmp/error.txt
-bash: /tmp/error.txt: cannot overwrite existing file
```

Para desactivar la opción `noclobber` para la sesión actual, ejecute `set +o noclobber` o `set +C`. Para que la opción `noclobber` sea persistente, debe incluirse en el perfil Bash del usuario o en el perfil de todo el sistema.

Incluso con la opción `noclobber` habilitada, es posible agregar datos redirigidos al contenido existente. Esto se logra con una redirección escrita con dos símbolos mayores que `>>`:

```
$ cat /proc/cpu_info 2>>/tmp/error.txt
$ cat /tmp/error.txt
cat: /proc/cpu_info: No such file or directory
cat: /proc/cpu_info: No such file or directory
```

En el ejemplo anterior, el nuevo mensaje de error se agregó al existente en el archivo `/tmp/error.txt`. Si el archivo aún no existe, se creará con los nuevos datos.

La fuente de datos de la entrada estándar de un proceso también se puede reasignar. El símbolo menor que `<` se usa para redirigir el contenido de un archivo al stdin de un proceso. En este caso, los datos fluyen de derecha a izquierda: se supone que el descriptor reasignado es 0 a la izquierda del símbolo menor que y el origen de datos (una ruta a un archivo) debe estar a la derecha del símbolo menor. El comando `uniq`, como la mayoría de las utilidades de línea de comandos para procesar texto, acepta los datos enviados a stdin por defecto:

```
$ uniq -c </tmp/error.txt
      2 cat: /proc/cpu_info: No such file or directory
```

La opción `-c` hace que `uniq` muestre cuántas veces aparece una línea repetida en el texto. Como se suprimió el valor numérico del descriptor de archivo redirigido, el comando de ejemplo es equivalente a `uniq -c 0</tmp/error.txt`. Usar un descriptor de archivo que no sea `0` en una redirección de entrada solo tiene sentido en contextos específicos, porque es posible que un programa solicite datos en los descriptores de archivo `3`, `4`, etc. De hecho, los programas pueden usar cualquier número entero por encima de 2 como nuevos descriptores de archivo para entrada/salida de datos. Por ejemplo, el siguiente código C lee los datos del descriptor de archivo `3` y simplemente lo replica en el descriptor de archivo `4`:

NOTE	El programa debe manejar dichos descriptores de archivo correctamente, de lo contrario podría intentar una operación de lectura o escritura no válida y bloquearse.

```
#include <stdio.h>

int main(int argc, char **argv){
  FILE *fd_3, *fd_4;
  // Open file descriptor 3
  fd_3 = fdopen(3, "r");
  // Open file descriptor 4
  fd_4 = fdopen(4, "w");
  // Read from file descriptor 3
  char buf[32];
  while ( fgets(buf, 32, fd_3) != NULL ){
    // Write to file descriptor 4
    fprintf(fd_4, "%s", buf);
  }
  // Close both file descriptors
  fclose(fd_3);
  fclose(fd_4);
}
```

Para probarlo, guarde el código de muestra como `fd.c` y compílelo con `gcc -o fd fd.c`. Este programa necesita que estén disponibles los descriptores de archivo 3 y 4 para poder leerlos y escribirlos. Como ejemplo, el archivo creado previamente `/tmp/error.txt` se puede utilizar como fuente del descriptor de archivo `3` y el descriptor de archivo `4` se puede redirigir a stdout:

```
$ ./fd 3</tmp/error.txt 4>&1
cat: /proc/cpu_info: No such file or directory
cat: /proc/cpu_info: No such file or directory
```

Desde la perspectiva del programador, el uso de descriptores de archivos evita tener que lidiar con el análisis de opciones y las rutas del sistema de archivos. El mismo descriptor de archivo puede incluso usarse como entrada y salida. En este caso, el descriptor de archivo se define en la línea de comandos con símbolos menores y mayores que, como en `3<>/tmp/error.txt`.

Here Document y Here String

Otra forma de redirigir la entrada involucra los métodos *Here Document* y *Here String*. La redirección de documentos *Here* permite escribir texto de varias líneas que se utilizará como contenido redirigido. Dos símbolos menor que `<<` indican una redirección de Here Document:

```
$ wc -c <<EOF
> How many characters
> in this Here document?
> EOF
43
```

A la derecha de los dos símbolos menor que `<<` se encuentra el término final `EOF`. El modo de inserción finalizará tan pronto como se ingrese una línea que contenga solo el término final. Se puede usar cualquier otro término como término final, pero es importante no poner caracteres en blanco entre el símbolo menor que y el término final. En el ejemplo anterior, las dos líneas de texto se enviaron al stdin del comando `wc -c`, que muestra el recuento de caracteres. Al igual que con los redireccionamientos de entrada para archivos, se supone el stdin (descriptor de archivo `0`) si se suprime el descriptor de archivo redirigido.

El método *Here String* es muy similar al método de *Here Document*, pero solo para una línea:

```
$ wc -c <<<"How many characters in this Here string?"
41
```

En este ejemplo, la cadena a la derecha de los tres signos menor que se envía al stdin de `wc -c`, que cuenta el número de caracteres. Las cadenas que contienen espacios deben estar entre comillas, de lo contrario, solo la primera palabra se usará como la cadena Here y las restantes se pasarán como argumentos al comando.

Ejercicios Guiados

1. Además de los archivos de texto, el comando `cat` también puede trabajar con datos binarios, como enviar el contenido de un dispositivo de bloque a un archivo. Usando la redirección, ¿cómo puede `cat` enviar el contenido del dispositivo `/dev/sdc` al archivo `sdc.img` en el directorio actual?

2. ¿Cuál es el nombre del canal estándar redirigido por el comando `date 1> now.txt`?

3. Después de intentar sobrescribir un archivo usando la redirección, un usuario recibe un error informando que la opción `noclobber` está habilitada. ¿Cómo se puede desactivar la opción `noclobber` para la sesión actual?

4. ¿Cuál será el resultado del comando `cat <<.>/dev/stdout`?

Ejercicios Exploratorios

1. El comando `cat /proc/cpu_info` muestra un mensaje de error porque `/proc/cpu_info` no existe. El comando `cat /proc/cpu_info 2>1` redirige el mensaje de error a dónde?

2. ¿Será posible descartar el contenido enviado a `/dev/null` si la opción `noclobber` está habilitada para la sesión de shell actual?

3. Sin usar `echo`, ¿cómo se podría redirigir el contenido de la variable `$USER` al stdin del comando `sha1sum`?

4. El kernel de Linux mantiene enlaces simbólicos en `/proc/PID/fd/` a cada archivo abierto por un proceso, donde *PID* es el número de identificación del proceso correspondiente. ¿Cómo podría el administrador del sistema usar ese directorio para verificar la ubicación de los archivos de registro abiertos por `nginx`, suponiendo que su PID sea `1234`?

5. Es posible hacer cálculos aritméticos utilizando solo comandos integrados de shell, pero los cálculos de coma flotante requieren programas específicos, como `bc` (*basic calculator*). Con `bc` incluso es posible especificar el número de lugares decimales, con el parámetro `scale`. Sin embargo, `bc` acepta operaciones solo a través de su entrada estándar, generalmente ingresada en modo interactivo. Usando una cadena Here, ¿cómo puede la operación de coma flotante `scale = 6; 1 / 3` enviarse a la entrada estándar de `bc`?

Resumen

Esta lección cubre los métodos para ejecutar un programa que redirige sus canales de comunicación estándar. Los procesos de Linux utilizan estos canales estándar como descriptores genéricos de archivos para leer y escribir datos, lo que permite cambiarlos arbitrariamente a archivos o dispositivos. La lección sigue los siguientes pasos:

- Qué son los descriptores de archivos y el papel que juegan en Linux.
- Los canales de comunicación estándar de cada proceso: *stdin*, *stdout* y *stderr*.
- Cómo ejecutar correctamente un comando utilizando la redirección de datos, tanto para entrada como para salida.
- Cómo usar *Here Document* y *Here String* en las redirecciones de entrada

Los comandos y procedimientos abordados fueron:

- Operadores de redireccionamiento: `>`, `<`, `>>`, `<<`, `<<<`.
- Comandos `cat`, `set`, `uniq` y `wc`.

Respuestas a los ejercicios guiados

1. Además de los archivos de texto, el comando `cat` también puede trabajar con datos binarios, como enviar el contenido de un dispositivo de bloque a un archivo. Usando la redirección, ¿cómo puede `cat` enviar el contenido del dispositivo `/dev/sdc` al archivo `sdc.img` en el directorio actual?

   ```
   $ cat /dev/sdc > sdc.img
   ```

2. ¿Cuál es el nombre del canal estándar redirigido por el comando `date 1> now.txt`?

   Salida estándar o stdout

3. Después de intentar sobrescribir un archivo usando la redirección, un usuario recibe un error informando que la opción `noclobber` está habilitada. ¿Cómo se puede desactivar la opción `noclobber` para la sesión actual?

   `set +C` o `set +o noclobber`

4. ¿Cuál será el resultado del comando `cat <<.>/dev/stdout`?

   Bash ingresará al modo de entrada Heredoc, luego saldrá cuando aparezca un punto en una línea por sí mismo. El texto escrito se redirigirá a stdout (impreso en la pantalla).

Respuestas a ejercicios exploratorios

1. El comando `cat /proc/cpu_info` muestra un mensaje de error porque `/proc/cpu_info` no existe. El comando `cat /proc/cpu_info 2>1` redirige el mensaje de error a dónde?

   A un archivo llamado `1` en el directorio actual.

2. ¿Será posible descartar el contenido enviado a `/dev/null` si la opción `noclobber` está habilitada para la sesión de shell actual?

   Si. `/dev/null` es un archivo especial no afectado por `noclobber`.

3. Sin usar `echo`, ¿cómo se podría redirigir el contenido de la variable `$USER` al stdin del comando `sha1sum`?

   ```
   $ sha1sum <<<$USER
   ```

4. El kernel de Linux mantiene enlaces simbólicos en `/proc/PID/fd/` a cada archivo abierto por un proceso, donde *PID* es el número de identificación del proceso correspondiente. ¿Cómo podría el administrador del sistema usar ese directorio para verificar la ubicación de los archivos de registro abiertos por `nginx`, suponiendo que su PID sea `1234`?

   Al emitir el comando `ls -l /proc/1234/fd`, que mostrará los objetivos de cada enlace simbólico en el directorio.

5. Es posible hacer cálculos aritméticos utilizando solo comandos integrados de shell, pero los cálculos de coma flotante requieren programas específicos, como `bc` (*basic calculator*). Con `bc` incluso es posible especificar el número de lugares decimales, con el parámetro `scale`. Sin embargo, `bc` acepta operaciones solo a través de su entrada estándar, generalmente ingresada en modo interactivo. Usando una cadena Here, ¿cómo puede la operación de coma flotante `scale = 6; 1 / 3` se enviará a la entrada estándar de `bc`?

   ```
   $ bc <<<"scale=6; 1/3"
   ```



103.4 Lección 2

Certificación:	LPIC-1 (101)
Versión:	5.0
Tema:	103 Comandos GNU y Unix
Objetivo:	103.4 Usar flujos, tuberías y redireccionamientos
Lección:	2 de 2

Introducción

Un aspecto de la filosofía de Unix establece que cada programa debe tener un propósito específico y no debe tratar de incorporar características fuera de su alcance. Pero mantener las cosas simples no significa resultados menos elaborados, ya que se pueden encadenar diferentes programas para producir un resultado combinado. El caracter de barra vertical `|`, también conocido como el símbolo *pipe*, se puede usar para crear una tubería que conecte la salida de un programa directamente a la entrada de otro, mientras que *command substitution* permite almacenar la salida de un programa en una variable o usarlo directamente como argumento para otro comando.

Tuberías (Pipes)

A diferencia de los redireccionamientos, con las tuberías los datos fluyen de izquierda a derecha en la línea de comandos y el objetivo es otro proceso, no una ruta del sistema de archivos, un descriptor de archivo o un documento. El carácter de canalización `|` le dice al shell que inicie todos los comandos distintos al mismo tiempo y que conecte la salida del comando anterior a la entrada del siguiente comando, de izquierda a derecha. Por ejemplo, en lugar de utilizar redireccionamientos, el contenido del archivo `/proc/cpuinfo` enviado a la salida estándar por `cat` puede canalizarse al stdin de `wc` con el siguiente comando:

```
$ cat /proc/cpuinfo | wc
    208    1184    6096
```

En ausencia de una ruta a un archivo, `wc` cuenta el número de líneas, palabras y caracteres que recibe en su stdin, como es el caso en el ejemplo. Muchas tuberías pueden estar presentes en un comando compuesto. En el siguiente ejemplo, se utilizan dos tuberías:

```
$ cat /proc/cpuinfo | grep 'model name' | uniq
model name      : Intel(R) Xeon(R) CPU           X5355  @ 2.66GHz
```

El contenido del archivo `/proc/cpuinfo` producido por `cat /proc/cpuinfo` se canalizó al comando `grep 'model name'`, que luego selecciona solo las líneas que contienen el término `model name`. La máquina que ejecuta el ejemplo tiene muchas CPU, por lo que hay líneas repetidas con `model name`. La última tubería conecta el nombre del modelo `grep 'model name'` a `uniq`, que es responsable de omitir cualquier línea igual a la anterior.

Las tuberías se pueden combinar con redireccionamientos en la misma línea de comando. El ejemplo anterior se puede reescribir en una forma más simple:

```
$ grep 'model name' </proc/cpuinfo | uniq
model name      : Intel(R) Xeon(R) CPU           X5355  @ 2.66GHz
```

La redirección de entrada para `grep` no es estrictamente necesaria ya que `grep` acepta una ruta de archivo como argumento, pero el ejemplo demuestra cómo construir dichos comandos combinados.

Las tuberías y redirecciones son exclusivas, es decir, una fuente puede asignarse a un solo destino. Sin embargo, es posible redirigir una salida a un archivo y aún verlo en la pantalla con el programa `tee`. Para hacerlo, el primer programa envía su salida al stdin de `tee` y se le proporciona un nombre de archivo a este último para almacenar los datos:

```
$ grep 'model name' </proc/cpuinfo | uniq | tee cpu_model.txt
model name      : Intel(R) Xeon(R) CPU           X5355  @ 2.66GHz
$ cat cpu_model.txt
model name      : Intel(R) Xeon(R) CPU           X5355  @ 2.66GHz
```

El resultado del último programa de la cadena, generado por `uniq`, se muestra y almacena en el archivo `cpu_model.txt`. Para no sobrescribir el contenido del archivo proporcionado sino para agregarle datos, la opción `-a` debe proporcionarse a `tee`.

Solo la salida estándar de un proceso es capturada por una tubería. Digamos que debe pasar por un largo proceso de compilación en la pantalla y al mismo tiempo guardar tanto la salida estándar como el error estándar en un archivo para su posterior inspección. Suponiendo que su directorio actual no tiene un *Makefile*, el siguiente comando generará un error:

```
$ make | tee log.txt
make: *** No targets specified and no makefile found.  Stop.
```

Aunque se muestra en la pantalla, el mensaje de error generado por `make` no fue capturado por `tee` y el archivo *log.txt* se creó vacío. Se debe hacer una redirección antes de que una tubería pueda capturar el stderr:

```
$ make 2>&1 | tee log.txt
make: *** No targets specified and no makefile found.  Stop.
$ cat log.txt
make: *** No targets specified and no makefile found.  Stop.
```

En este ejemplo, el stderr de `make` se redirigió al stdout, por lo que `tee` pudo capturarlo con una tubería, mostrarlo en la pantalla y guardarlo en el archivo `log.txt`. En casos como este, puede ser útil guardar los mensajes de error para su posterior inspección.

Sustitución de comando

Otro método para capturar la salida de un comando es *command substitution* (sustitución de comando). Al colocar un comando dentro de las comillas inversas, Bash lo reemplaza con su salida estándar. El siguiente ejemplo muestra cómo usar el stdout de un programa como argumento para otro programa:

```
$ mkdir `date +%Y-%m-%d`
$ ls
2019-09-05
```

La salida del programa `date`, la fecha actual formateada como *año-mes-día*, se utilizó como argumento para crear un directorio con `mkdir`. Se obtiene un resultado idéntico usando `$()` en lugar de comillas inversas:

```
$ rmdir 2019-09-05
$ mkdir $(date +%Y-%m-%d)
$ ls
2019-09-05
```

El mismo método puede usarse para almacenar la salida de un comando como una variable:

```
$ OS=`uname -o`
$ echo $OS
GNU/Linux
```

El comando `uname -o` genera el nombre genérico del sistema operativo actual, que estaba almacenado en la variable de sesión `OS`. Asignar la salida de un comando a una variable es muy útil en los scripts, ya que permite almacenar y evaluar los datos de muchas maneras distintas.

Dependiendo de la salida generada por el comando reemplazado, la sustitución del comando incorporado puede no ser apropiada. Un método más sofisticado para usar la salida de un programa como argumento de otro programa emplea un intermediario llamado `xargs`. El programa `xargs` usa los contenidos que recibe a través de stdin para ejecutar un comando dado con los contenidos como argumento. El siguiente ejemplo muestra `xargs` ejecutando el programa `identify` con argumentos proporcionados por el programa `find`:

```
$ find /usr/share/icons -name 'debian*' | xargs identify -format "%f: %wx%h\n"
debian-swirl.svg: 48x48
debian-swirl.png: 22x22
debian-swirl.png: 32x32
debian-swirl.png: 256x256
debian-swirl.png: 48x48
debian-swirl.png: 16x16
debian-swirl.png: 24x24
debian-swirl.svg: 48x48
```

El programa `identify` es parte de *ImageMagick*, un conjunto de herramientas de línea de comandos para inspeccionar, convertir y editar la mayoría de los tipos de archivos de imagen. En el ejemplo, `xargs` tomó todas las rutas listadas por `find` y las puso como argumentos para `identify`, que luego muestra la información para cada archivo formateado como lo requiere la opción `-format`. Los archivos encontrados por `find` en el ejemplo son imágenes que contienen el logotipo de distribución en un sistema de archivos Debian. `-format` es un parámetro para `identify`, no para `xargs`.

La opción `-n 1` requiere que `xargs` ejecute el comando dado con un solo argumento a la vez. En el

caso del ejemplo, en lugar de pasar todas las rutas encontradas por `find` como una lista de argumentos para `identify`, usar `xargs -n 1` ejecutaría el comando `identify` para cada ruta por separado. El uso de `-n 2` ejecutaría `identify` con dos rutas como argumentos, `-n 3` con tres rutas como argumentos y así sucesivamente. De manera similar, cuando `xargs` procesa contenidos de varias líneas, como es el caso de la entrada proporcionada por `find`, la opción `-L` puede usarse para limitar cuántas líneas se usarán como argumentos por ejecución de comando.

NOTE

Usar `xargs` con la opción `-n 1` o `-L 1` para procesar la salida generada por `find` puede ser innecesario El comando `find` tiene la opción `-exec` para ejecutar un comando dado para cada elemento de resultado de búsqueda.

Si las rutas tienen caracteres de espacio, es importante ejecutar `find` con la opción `-print0`. Esta opción indica a `find` que use un carácter nulo entre cada entrada para que la lista pueda ser analizada correctamente por `xargs` (se suprimió la salida):

```
$ find . -name '*avi' -print0 -o -name '*mp4' -print0 -o -name '*mkv' -print0 |
xargs -0 du | sort -n
```

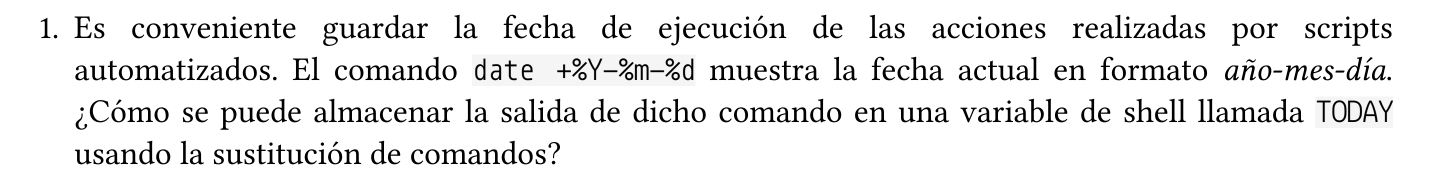
La opción `-0` le dice a `xargs` que el caracter nulo debe usarse como separador. De esa manera, las rutas de archivo proporcionadas por `find` se analizan correctamente incluso si tienen caracteres en blanco u otros caracteres especiales. El ejemplo anterior muestra cómo usar el comando `du` para averiguar el espacio ocupado en disco de cada archivo encontrado y luego ordenar los resultados por tamaño. La salida fue suprimida por concisión. Tenga en cuenta que para cada criterio de búsqueda es necesario colocar la opción `-print0` para `find`.

Por defecto, `xargs` coloca los argumentos del comando ejecutado en último lugar. Para cambiar ese comportamiento, se debe usar la opción `-I`:

```
$ find . -mindepth 2 -name '*avi' -print0 -o -name '*mp4' -print0 -o -name '*mkv'
-print0 | xargs -0 -I PATH mv PATH ./
```

En el último ejemplo, cada archivo encontrado por `find` se mueve al directorio actual. Como las rutas de origen deben ser informadas a `mv` antes de la ruta de destino, se da un término de sustitución a la opción `-I` de `xargs` que luego se coloca apropiadamente junto a `mv`. Al utilizar el caracter nulo como separador, no es necesario encerrar el término de sustitución con comillas.

Ejercicios Guiados



1. Es conveniente guardar la fecha de ejecución de las acciones realizadas por scripts automatizados. El comando `date +%Y-%m-%d` muestra la fecha actual en formato *año-mes-día*. ¿Cómo se puede almacenar la salida de dicho comando en una variable de shell llamada `TODAY` usando la sustitución de comandos?

2. Usando el comando `echo`, ¿cómo se puede enviar el contenido de la variable `TODAY` a la entrada estándar del comando `sed s/-/./g`?

3. ¿Cómo podría usarse la salida del comando `date +%Y-%m-%d` como una cadena Here para ordenar `sed s/-/./g`?

4. El comando `convert image.jpeg -resize 25% small/image.jpeg` crea una versión más pequeña de `image.jpeg` y coloca la imagen resultante en un archivo con el mismo nombre dentro del subdirectorio `small`. Usando `xargs`, ¿cómo es posible ejecutar el mismo comando para cada imagen listada en el archivo `filelist.txt`?

Ejercicios Exploratorios

1. Una rutina de copia de seguridad simple crea periódicamente una imagen de partición `/dev/sda1` con `dd < /dev/sda1 > sda1.img`. Para realizar futuras comprobaciones de integridad de datos, la rutina también genera un hash SHA1 del archivo con `sha1sum < sda1.img > sda1.sha1`. Al agregar tuberías y el comando `tee`, ¿cómo se combinarían estos dos comandos en uno?

2. El comando `tar` se usa para archivar muchos archivos en uno solo, preservando la estructura del directorio. La opción `-T` permite especificar un archivo que contiene las rutas a archivar. Por ejemplo, `find /etc -type f | tar -cJ -f /srv/backup/etc.tar.xz -T -` crea un archivo tar comprimido `etc.tar.xz` de la lista provista por el comando `find` (la opción `-T-` indica la entrada estándar como la lista de ruta). Para evitar posibles errores de análisis debido a las rutas que contienen espacios, ¿qué opciones de comando deberían estar presentes para `find` y `tar`?

3. En lugar de abrir una nueva sesión de shell remota, el comando `ssh` solo puede ejecutar un comando indicado como argumento: `ssh user@storage "remote command"`. Dado que `ssh` también permite redirigir la salida estándar de un programa local a la entrada estándar del programa remoto, ¿cómo canalizaría el comando `cat` un archivo local llamado `etc.tar.gz` a `/srv/backup/etc.tar.gz` en `user@storage` a través de `ssh`?

Resumen

Esta lección cubre las técnicas tradicionales de comunicación entre procesos empleadas por Linux. *La canalización de comandos* crea un canal de comunicación unidireccional entre dos procesos y la *sustitución de comandos* permite almacenar la salida de un proceso en una variable de shell. La lección sigue los siguientes pasos:

- Cómo se pueden usar *pipes* para transmitir la salida de un proceso a la entrada de otro.
- El propósito de los comandos `tee` y `xargs`.
- Cómo capturar el resultado de un proceso con *command sustitution*, almacenándolo en una variable o usándolo directamente como parámetro para otro comando.

Los comandos y procedimientos abordados fueron:

- Comando de canalización con `|`.
- Sustitución de comandos con backticks y `$()`.
- Comandos `tee`, `xargs` y `find`.

Respuestas a los ejercicios guiados

1. Es conveniente guardar la fecha de ejecución de las acciones realizadas por scripts automatizados. El comando `date +%Y-%m-%d` muestra la fecha actual en formato *año-mes-día*. ¿Cómo se puede almacenar la salida de dicho comando en una variable de shell llamada `TODAY` usando la sustitución de comandos?

   ```
   $ TODAY=`date +%Y-%m-%d`
   ```

   o

   ```
   $ TODAY=$(date +%Y-%m-%d)
   ```

2. Usando el comando `echo`, ¿cómo se puede enviar el contenido de la variable `TODAY` a la entrada estándar del comando `sed s/-/./g`?

   ```
   $ echo $TODAY | sed s/-/./g
   ```

3. ¿Cómo podría usarse la salida del comando `date +%Y-%m-%d` como una cadena Here para ordenar `sed s/-/./g`?

   ```
   $ sed s/-/./g <<< `date +%Y-%m-%d`
   ```

   o

   ```
   $ sed s/-/./g <<< $(date +%Y-%m-%d)
   ```

4. El comando `convert image.jpeg -resize 25% small/image.jpeg` crea una versión más pequeña de `image.jpeg` y coloca la imagen resultante en un archivo con el mismo nombre dentro del subdirectorio `small`. Usando `xargs`, ¿cómo es posible ejecutar el mismo comando para cada imagen listada en el archivo` filelist.txt`?

   ```
   $ xargs -I IMG convert IMG -resize 25% small/IMG < filelist.txt
   ```

   o

```
$ cat filelist.txt | xargs -I IMG convert IMG -resize 25% small/IMG
```

Respuestas a ejercicios exploratorios

1. Una rutina de copia de seguridad simple crea periódicamente una imagen de partición `/dev/sda1` con `dd < /dev/sda1 > sda1.img`. Para realizar futuras comprobaciones de integridad de datos, la rutina también genera un hash SHA1 del archivo con `sha1sum < sda1.img > sda1.sha1`. Al agregar tuberías y el comando `tee`, ¿cómo se combinarían estos dos comandos en uno?

   ```
   # dd < /dev/sda1 | tee sda1.img | sha1sum > sda1.sha1
   ```

2. El comando `tar` se usa para archivar muchos archivos en uno solo, preservando la estructura del directorio. La opción `-T` permite especificar un archivo que contiene las rutas a archivar. Por ejemplo, `find /etc -type f | tar -cJ -f /srv/backup/etc.tar.xz -T -` crea un archivo tar comprimido `etc.tar.xz` de la lista provista por el comando `find` (la opción `-T-` indica la entrada estándar como la lista de ruta). Para evitar posibles errores de análisis debido a las rutas que contienen espacios, ¿qué opciones de comando deberían estar presentes para `find` y `tar`?

   Opciones `-print0` y `--null`:

   ```
   $ find /etc -type f -print0 | tar -cJ -f /srv/backup/etc.tar.xz --null -T -
   ```

3. En lugar de abrir una nueva sesión de shell remota, el comando `ssh` solo puede ejecutar un comando indicado como argumento: `ssh user@storage "remote command"`. Dado que `ssh` también permite redirigir la salida estándar de un programa local a la entrada estándar del programa remoto, ¿cómo canalizaría el comando `cat` un archivo local llamado `etc.tar.gz` a `/srv/backup/etc.tar.gz` en `user@storage` a través de `ssh`?

   ```
   $ cat etc.tar.gz | ssh user@storage "cat > /srv/backup/etc.tar.gz"
   ```

   o

   ```
   $ ssh user@storage "cat > /srv/backup/etc.tar.gz" < etc.tar.gz
   ```



103.5 Crear, supervisar y matar procesos

Referencia al objetivo del LPI

LPIC-1 v5, Exam 101, Objective 103.5

Importancia

4

Áreas de conocimiento clave

- Ejecutar trabajos en primer y segundo plano.
- Enviar señales a los programas para que continúen ejecutándose después del cierre de sesión.
- Supervisar procesos activos.
- Seleccionar y ordenar procesos para su visualización.
- Enviar señales a los procesos.

Lista parcial de archivos, términos y utilidades

- `&`
- `bg`
- `fg`
- `jobs`
- `kill`
- `nohup`
- `ps`
- `top`

- `free`
- `uptime`
- `pgrep`
- `pkill`
- `killall`
- `watch`
- `screen`
- `tmux`



103.5 Lección 1

Certificación:	LPIC-1
Versión:	5.0
Tema:	103 Comandos GNU y Unix
Objetivo:	103.5 Crear, monitorear y matar procesos
Lección:	1 de 2

Introducción

Cada vez que invocamos un comando, se inician uno o más procesos. Un administrador de sistemas bien entrenado no solo necesita crear procesos, sino también poder realizar un seguimiento de ellos y enviarles diferentes tipos de señales si es necesario. En esta lección veremos el control del trabajo y cómo monitorear los procesos.

Control de trabajos

Los *trabajos* (Jobs) son procesos que se han iniciado de forma interactiva a través de un terminal, enviados a un segundo plano y aún no han finalizado la ejecución. Puede conocer los trabajos activos (y su estado) en su sistema Linux ejecutando `jobs`:

```
$ jobs
```

El comando `jobs` anterior no produjo ningún resultado, lo que significa que no hay trabajos activos en este momento. Creemos nuestro primer trabajo ejecutando un comando que tarde un poco en finalizar la ejecución (el comando `sleep` con un parámetro de `60`) y, mientras se ejecuta, presione `Ctrl` + `Z`:

```
$ sleep 60
^Z
[1]+  Stopped                 sleep 60
```

La ejecución del comando se ha detenido (o, mejor dicho, suspendido) y el símbolo del sistema vuelve a estar disponible. Puede buscar trabajos por segunda vez y encontrará el *suspendido*:

```
$ jobs
[1]+  Stopped                 sleep 60
```

Permítanos explicar el resultado:

`[1]`

Este número es el ID del trabajo y se puede utilizar, precedido por un símbolo de porcentaje (`%`), para cambiar el estado del trabajo mediante las utilidades `fg`, `bg` y `kill` (como se mostrará más adelante).

`+`

El signo más indica el trabajo actual predeterminado (es decir, el último suspendido o enviado al segundo plano). El trabajo anterior está marcado con un signo menos (`-`). Cualquier otro trabajo anterior no está marcado.

`Stopped`

Descripción del estado del trabajo.

`sleep 60`

El comando o trabajo en ejecución.

Con la opción `-l`, los trabajos también mostrarán la ID del proceso (PID) justo antes del estado:

```
$ jobs -l
[1]+  1114 Stopped                 sleep 60
```

Las opciones posibles restantes de trabajos son:

`-n`

Lista solo los procesos que han cambiado de estado desde la última notificación. El estado posible incluye, `Running`, `Stopped`, `Terminated` o `Done`.

`-p`
: Lista los IDs de procesos.

`-r`
: Lista solo los procesos en ejecución.

`-s`
: Lista solamente los trabajos detenidos (o suspendidos).

NOTE Recuerde, un trabajo tiene un *ID de trabajo* y un *ID de proceso* (PID).

Especificaciones de trabajos

El comando `jobs`, así como otras utilidades como `fg`, `bg` y `kill` (que verá en la siguiente sección) necesitan una especificación de trabajo (o `jobspec`) para actuar sobre un trabajo en particular. Como acabamos de ver, esto puede ser, y normalmente es, el ID del trabajo precedido por `%`. Sin embargo, otras especificaciones de trabajo también son posibles. Echemos un vistazo a ellos:

`%n`
: Trabajo cuyo número de identificación es `n`:

```
$ jobs %1
[1]+  Stopped                 sleep 60
```

`%str`
: Trabajo cuya línea de comando comienza con `str`:

```
$ jobs %sl
[1]+  Stopped                 sleep 60
```

`%?str`
: Trabajo cuya línea de comando contiene `str`:

```
$ jobs %?le
[1]+  Stopped                 sleep 60
```

%+ o %%

Trabajo actual (el último que se inició en segundo plano o suspendido del primer plano):

```
$ jobs %+
[1]+  Stopped                 sleep 60
```

%-

Trabajo anterior (el que era `% +` antes del predeterminado, el actual):

```
$ jobs %-
[1]+  Stopped                 sleep 60
```

En nuestro caso, dado que solo hay un trabajo, es actual y anterior.

Estado del trabajo: suspensión, primer plano y segundo plano

Una vez que un trabajo está en segundo plano o ha sido suspendido, podemos hacer cualquiera de estas tres cosas:

1. Llevarlo al primer plano con `fg`:

```
$ fg %1
sleep 60
```

`fg` mueve el trabajo especificado al primer plano y lo convierte en el trabajo actual. Ahora podemos esperar hasta que termine, detenerlo nuevamente con Ctrl + Z o terminarlo con Ctrl + C.

2. Llevarlo a un segundo plano con `bg`:

```
$ bg %1
[1]+ sleep 60 &
```

Una vez en segundo plano, el trabajo se puede volver a poner en primer plano con `fg` o matar (ver más abajo). Tenga en cuenta el signo (&) que significa que el trabajo se ha enviado a segundo plano. De hecho, también puede usar el signo y comenzar un proceso directamente en segundo plano:

```
$ sleep 100 &
[2] 970
```

Junto con el ID de trabajo del nuevo trabajo (`[2]`), ahora también obtenemos su ID de proceso (`970`). Ahora ambos trabajos se ejecutan en segundo plano:

```
$ jobs
[1]-  Running                 sleep 60 &
[2]+  Running                 sleep 100 &
```

Un poco más tarde, el primer trabajo finaliza la ejecución:

```
$ jobs
[1]-  Done                    sleep 60
[2]+  Running                 sleep 100 &
```

3. Termine con una señal `SIGTERM` con `kill`:

```
$ kill %2
```

Para asegurarse de que el trabajo ha finalizado, ejecute `jobs` nuevamente:

```
$ jobs
[2]+  Terminated                 sleep 100
```

NOTE Si no se especifica ningún trabajo, `fg` y `bg` actuarán sobre el actual, predeterminado. `kill`, sin embargo, siempre necesita una especificación de trabajo.

Trabajos separados: `nohup`

Los trabajos que hemos visto en las secciones anteriores se adjuntaron a la sesión del usuario que los invocó. Eso significa que si la sesión se termina, los trabajos desaparecen. Sin embargo, es posible separar los trabajos de las sesiones y hacer que se ejecuten incluso después de cerrar la sesión. Esto se logra con el comando `nohup` ("no hangup"). La sintaxis es la siguiente:

```
nohup COMMAND &
```

Recuerde, el `&` envía el proceso a un segundo plano y libera el terminal en el que está trabajando.

Separemos el trabajo en segundo plano `ping localhost` de la sesión actual:

```
$ nohup ping localhost &
[1] 1251
$ nohup: ignoring input and appending output to 'nohup.out'
^C
```

La salida nos muestra la ID del trabajo (`[1]`) y el PID (`1251`), seguido de un mensaje que nos informa sobre el archivo `nohup.out`. Este es el archivo predeterminado donde se guardarán `stdout` y `stderr`. Ahora podemos presionar `Ctrl`+`C` para liberar el símbolo del sistema, cerrar la sesión, iniciar otro como `root` y usar `tail -f` para verificar si el comando se está ejecutando y la salida se está escribiendo en el archivo predeterminado:

```
$ exit
logout
$ tail -f /home/carol/nohup.out
64 bytes from localhost (::1): icmp_seq=3 ttl=64 time=0.070 ms
64 bytes from localhost (::1): icmp_seq=4 ttl=64 time=0.068 ms
64 bytes from localhost (::1): icmp_seq=5 ttl=64 time=0.070 ms
^C
```

TIP En lugar de utilizar el `nohup.out` predeterminado, podría haber especificado el archivo de salida de su elección con `nohup ping localhost > /path/to/your/file &`.

Si queremos matar el proceso, debemos especificar su PID:

```
# kill 1251
```

Monitoreo de procesos

Un proceso o tarea es una instancia de un programa en ejecución. Por lo tanto, se crean nuevos procesos cada vez que escribe comandos en el terminal.

El comando `watch` ejecuta un programa periódicamente (2 segundos por defecto) y nos permite *mirar*

el cambio de salida del programa con el tiempo. Por ejemplo, podemos monitorear cómo cambia el promedio de carga a medida que se ejecutan más procesos escribiendo `watch uptime`:

```
Every  2.0s: uptime          debian: Tue Aug 20 23:31:27 2019

 23:31:27 up 21 min,  1 user,  load average: 0.00, 0.00, 0.00
```

El comando se ejecuta hasta que se interrumpe, por lo que deberíamos detenerlo con `Ctrl`+`C`. Obtenemos dos líneas como salida: la primera corresponde a `watch` y nos dice con qué frecuencia se ejecutará el comando (`Every 2.0s: uptime`), qué comando/programa mirar (`uptime`) así como el comando nombre de host y fecha (`debian: mar 20 de agosto 23:31:27 2019`). La segunda línea de salida es el tiempo de actividad e incluye la hora (`23:31:27`), cuánto tiempo ha estado activo el sistema (`up 21 min`), el número de usuarios activos (`1 usuario`) y carga promedio del sistema o número de procesos en ejecución o en estado de espera durante los últimos 1, 5 y 15 minutos (`promedio de carga: 0.00, 0.00, 0.00`).

Del mismo modo, puede verificar el uso de la memoria a medida que se crean nuevos procesos con `watch free`:

```
Every  2.0s: free            debian: Tue Aug 20 23:43:37 2019

 23:43:37 up 24 min,  1 user,  load average: 0.00, 0.00, 0.00
              total        used        free      shared  buff/cache   available
Mem:       16274868      493984    14729396       35064     1051488    15462040
Swap:      16777212           0    16777212
```

Para cambiar el intervalo de actualización para `watch` use las opciones `-n` o `--interval` más el número de segundos como en:

```
$ watch -n 5 free
```

Ahora el comando `free` se ejecutará cada 5 segundos.

Para obtener más información sobre las opciones de `uptime`, `free` y `watch`, consulte sus páginas de manual.

NOTE La información proporcionada por `uptime` y `free` también está integrada en las herramientas más completas `top` y `ps` (ver más abajo).

Envío de señales a procesos: `kill`

Cada proceso tiene un identificador de proceso único o PID. Una forma de averiguar el PID de un proceso es mediante el comando `pgrep` seguido del nombre del proceso:

```
$ pgrep sleep
1201
```

NOTE El identificador de un proceso también se puede descubrir a través del comando `pidof` (por ejemplo, `pidof sleep`).

Similar a `pgrep`, el comando `pkill` mata un proceso basado en su nombre:

```
$ pkill sleep
[1]+  Terminated                 sleep 60
```

Para matar varias instancias del mismo proceso, se puede usar el comando `killall`:

```
$ sleep 60 &
[1] 1246
$ sleep 70 &
[2] 1247
$ killall sleep
[1]-  Terminated                 sleep 60
[2]+  Terminated                 sleep 70
```

Tanto `pkill` como `killall` funcionan de la misma manera que `kill` en que envían una señal de terminación a los procesos especificados. Si no se proporciona ninguna señal, se envía el valor predeterminado de `SIGTERM`. Sin embargo, `kill` solo toma un trabajo o una ID de proceso como argumento.

Las señales se pueden especificar por:

- Nombre:

  ```
  $ kill -SIGHUP 1247
  ```

- Número:

```
$ kill -1 1247
```

- Opciones:

```
$ kill -s SIGHUP 1247
```

Para que `kill` funcione de manera similar a `pkill` o `killall` (y nos ahorremos los comandos para descubrir los PID primero) podemos usar la sustitución de comandos:

```
$ kill -1 $(pgrep sleep)
```

Como ya debería saber, una sintaxis alternativa es `` kill -1 `pgrep sleep` ``.

> **TIP** Para obtener una lista exhaustiva de todas las señales `kill` y sus códigos, escriba `kill -l` en el terminal. Use `-KILL` (`-9` o `-s KILL`) para matar los procesos rebeldes cuando falla cualquier otra señal.

`top` **y** `ps`

Cuando se trata de monitoreo de procesos, dos herramientas invaluables son `top` y `ps`. Mientras que el primero produce resultados dinámicamente, el segundo lo hace estáticamente. En cualquier caso, ambos son excelentes utilidades para tener una visión integral de todos los procesos en el sistema.

Interactuando con `top`

Para invocar `top`, simplemente teclee `top`:

```
$ top

top - 11:10:29 up  2:21,  1 user,  load average: 0,11, 0,20, 0,14
Tasks:  73 total,   1 running,  72 sleeping,   0 stopped,   0 zombie
%Cpu(s):  0,0 us,  0,3 sy,  0,0 ni, 99,7 id,  0,0 wa,  0,0 hi,  0,0 si,  0,0 st
KiB Mem :  1020332 total,   909492 free,    38796 used,    72044 buff/cache
KiB Swap:  1046524 total,  1046524 free,        0 used.   873264 avail Mem

   PID USER      PR  NI    VIRT    RES    SHR S %CPU %MEM     TIME+ COMMAND
   436 carol     20   0   42696   3624   3060 R  0,7  0,4   0:00.30 top
     4 root      20   0       0      0      0 S  0,3  0,0   0:00.12 kworker/0:0
   399 root      20   0   95204   6748   5780 S  0,3  0,7   0:00.22 sshd
     1 root      20   0   56872   6596   5208 S  0,0  0,6   0:01.29 systemd
     2 root      20   0       0      0      0 S  0,0  0,0   0:00.00 kthreadd
     3 root      20   0       0      0      0 S  0,0  0,0   0:00.02 ksoftirqd/0
     5 root       0 -20       0      0      0 S  0,0  0,0   0:00.00 kworker/0:0H
     6 root      20   0       0      0      0 S  0,0  0,0   0:00.00 kworker/u2:0
     7 root      20   0       0      0      0 S  0,0  0,0   0:00.08 rcu_sched
     8 root      20   0       0      0      0 S  0,0  0,0   0:00.00 rcu_bh
     9 root      rt   0       0      0      0 S  0,0  0,0   0:00.00 migration/0
    10 root       0 -20       0      0      0 S  0,0  0,0   0:00.00 lru-add-drain
    (...)
```

`top` le permite al usuario cierta interacción. Por defecto, la salida se ordena por el porcentaje de tiempo de CPU utilizado por cada proceso en orden descendente. Este comportamiento puede modificarse presionando las siguientes teclas desde `top`:

`M`

Odena por uso de *memoria*.

`N`

Ordena por *número* de ID.

`T`

Ordena por *tiempo* de ejecución.

`P`

Ordena por *porcentaje* de uso en CPU.

TIP Para cambiar entre orden descendente/ascendente, simplemente presione `R`.

Otras teclas interesantes para interactuar con `top` son:

`?` o `h`
: Ayuda.

`k`
: Mata un proceso. `top` solicitará que se elimine el `PID` del proceso y que se envíe la señal (por defecto, `SIGTERM` o `15`).

`r`
: Cambiar la prioridad de un proceso (`renice`). `top` le pedirá el valor `nice`. Los valores posibles oscilan entre -20 y 19, pero solo el superusuario (`root`) puede establecerlo en un valor negativo o inferior al actual.

`u`
: Lista de procesos de un usuario en particular (de forma predeterminada se muestran los procesos de todos los usuarios).

`c`
: Muestra las rutas absolutas de los programas y diferencia entre procesos de espacio de usuario y procesos de espacio de kernel (entre corchetes).

`V`
: Vista de bosque/jerarquía de procesos.

`t` y `m`
: Cambia el aspecto de las lecturas de CPU y memoria respectivamente en un ciclo de cuatro etapas: las dos primeras pulsaciones muestran barras de progreso, la tercera oculta la barra y la cuarta la recupera.

`W`
: Guardar ajustes de configuración en `~/.toprc`.

TIP Una versión más elegante y fácil de usar de `top` es `htop`. Otra alternativa, quizás más exhaustiva, es `atop`. Si aún no está instalado en su sistema, use su administrador de paquetes para instalarlos y probarlos.

Una explicación de la salida de `top`

La salida `top` se divide en dos áreas: el *área resumen* y el *área de tareas*.

El área de resumen en `top`

El *área de resumen* se compone de las cinco filas superiores y nos proporciona la siguiente información:

- `top - 11:10:29 up 2:21, 1 user, load average: 0,11, 0,20, 0,14`
  - Hora actual (formato 24 horas): `11:20:29`
  - Tiempo de actividad (cantidad de tiempo que el equipo ha estado activo y funcionando): `up 2:21`
  - Número de usuarios conectados y promedio de carga de la CPU durante los últimos 1, 5 y 15 minutos, respectivamente: `load average: 0,11, 0,20, 0,14`
- `Tasks: 73 total, 1 running, 72 sleeping, 0 stopped, 0 zombie` (información sobre procesos)
  - Número total de procesos en modo activo: `73 total`
  - Ejecutándose (los ejecutados en el momento): `1 running`
  - Durmiendo (aquellos que esperan reanudar la ejecución): `72 sleeping`
  - Detenido (por una señal de control de trabajo): `0 stopped`
  - Zombie (aquellos que han completado la ejecución pero todavía están esperando que su proceso padre los elimine de la tabla de procesos): `0 zombie`
- `%Cpu(s): 0,0 us, 0,3 sy, 0,0 ni, 99,7 id, 0,0 wa, 0,0 hi, 0,0 si, 0,0 st` (porcentaje de tiempo de CPU empleado)
  - Procesos de usuario: `0,0 us`
  - Procesos de sistema/kernel: `0,4 sy`
  - Procesos establecidos en un valor *nice*—cuanto mejor sea el valor, menor será la prioridad: `0,0 ni`
  - Nada—tiempo de inactividad de la CPU: `99,7 id`
  - Procesos en espera de operaciones de E/S: `0,0 wa`
  - Procesos que sirven interrupciones de hardware—periféricos que envían las señales del procesador que requieren atención: `0,0 hi`
  - Procesos que sirven interrupciones de software: `0,0 si`

   - Los procesos que sirven las tareas de otras máquinas virtuales en un entorno virtual, por lo tanto, roban tiempo: `0,0 st`

- `KiB Mem : 1020332 total, 909492 free, 38796 used, 72044 buff/cache` (Información de memoria en kilobytes)
   - Monto total de memoria: `1020332 total`
   - Memoria sin utilizar: `909492 free`
   - Memoria en uso: `38796 used`
   - La memoria intermedia (buffer) y almacenada en caché para evitar el acceso excesivo al disco: `72044 buff/cache`

     Observe cómo el `total` es la suma de los otros tres valores — `free`, `used` y `buff/cache` — (aproximadamente 1 GB en nuestro caso).

- `KiB Swap: 1046524 total, 1046524 free, 0 used. 873264 avail Mem` (Información memoria swap en kilobytes)
   - La cantidad total de espacio de swap: `1046524 total`
   - Espacio de swap no utilizado: `1046524 free`
   - Espacio en uso de swap: `0 used`
   - La cantidad de memoria de intercambio que se puede asignar a los procesos sin causar más intercambio: `873264 avail Mem`

El área de tareas en `top`: campos y columnas

Debajo del área de resumen, aparece el área de tareas, que incluye una serie de campos y columnas de información sobre los procesos en ejecución:

`PID`
: Identificador de proceso.

`USER`
: Usuario que emitió el comando que generó el proceso.

`PR`
: Prioridad de proceso en el kernel.

`NI`

Valor nice del proceso. Los valores más bajos tienen mayor prioridad que los más altos.

`VIRT`

Cantidad total de memoria utilizada por el proceso (incluido la swap).

`RES`

Memoria RAM utilizada por el proceso.

`SHR`

Memoria compartida del proceso con otros procesos.

`S`

Estado del proceso. Los valores incluyen: `S` (suspensión interrumpible — esperando que termine un evento), `R` (ejecutable — ya sea en ejecución o en la cola que se ejecutará) o `Z` (procesos secundarios terminados en zombies cuyas estructuras de datos aún no se han eliminado de la tabla de procesos).

`%CPU`

Porcentaje de CPU utilizado por el proceso.

`%MEM`

Porcentaje de RAM utilizada por el proceso, es decir, el valor `RES` expresado como porcentaje.

`TIME+`

Tiempo total de actividad del proceso.

`COMMAND`

Nombre del comando/programa que generó el proceso..

Visualización de procesos estáticos: `ps`

Como se dijo anteriormente, `ps` muestra una instantánea de los procesos. Para ver todos los procesos con un terminal (tty), escriba `ps a`:

```
$ ps a
  PID TTY      STAT   TIME COMMAND
  386 tty1     Ss+    0:00 /sbin/agetty --noclear tty1 linux
  424 tty7     Ssl+   0:00 /usr/lib/xorg/Xorg :0 -seat seat0 (...)
  655 pts/0    Ss     0:00 -bash
 1186 pts/0    R+     0:00 ps a
 (...)
```

Una explicación de la sintaxis y salida de la opción `ps`

Con respecto a las opciones, `ps` puede aceptar tres estilos diferentes: BSD, UNIX y GNU. Veamos cómo funcionaría cada uno de estos estilos al informar información sobre un ID de proceso en particular:

BSD

Las opciones no siguen ningún guión inicial:

```
$ ps p 811
  PID TTY      STAT   TIME COMMAND
  811 pts/0    S      0:00 -su
```

UNIX

Las opciones siguen un guión inicial:

```
$ ps -p 811
  PID TTY          TIME CMD
  811 pts/0    00:00:00 bash
```

GNU

Las opciones van seguidas de guiones dobles iniciales:

```
$ ps --pid 811
  PID TTY          TIME CMD
  811 pts/0    00:00:00 bash
```

En los tres casos, `ps` informa sobre el proceso cuyo `PID` es `811` — en este caso, `bash`.

Del mismo modo, puede usar `ps` para buscar los procesos iniciados por un usuario en particular:

- `ps U carol` (BSD)
- `ps -u carol` (UNIX)
- `ps --user carol` (GNU)

Veamos los procesos iniciados por `carol`:

```
$ ps U carol
  PID TTY      STAT   TIME COMMAND
  811 pts/0    S      0:00 -su
  898 pts/0    R+     0:00 ps U carol
```

Comenzó dos procesos: `bash` (`-su`) y `ps` (`ps U carol`). La columna `STAT` nos dice el estado del proceso (ver más abajo).

Podemos obtener lo mejor de `ps` combinando algunas de sus opciones. Un comando muy útil (que produce una salida similar a la de `top`) es `ps aux` (estilo BSD). En este caso, se muestran los procesos de todos los shells (no solo el actual). El significado de los interruptores es el siguiente:

`a`

Mostrar procesos que están conectados a un `tty` o terminal.

`u`

Mostrar formato orientado al usuario.

`x`

Mostrar procesos que no están conectados a un `tty` o terminal.

```
$ ps aux
USER       PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root         1  0.0  0.1 204504  6780 ?        Ss   14:04   0:00 /sbin/init
root         2  0.0  0.0      0     0 ?        S    14:04   0:00 [kthreadd]
root         3  0.0  0.0      0     0 ?        S    14:04   0:00 [ksoftirqd/0]
root         5  0.0  0.0      0     0 ?        S<   14:04   0:00 [kworker/0:0H]
root         7  0.0  0.0      0     0 ?        S    14:04   0:00 [rcu_sched]
root         8  0.0  0.0      0     0 ?        S    14:04   0:00 [rcu_bh]
root         9  0.0  0.0      0     0 ?        S    14:04   0:00 [migration/0]
(...)
```

Permítanos explicar las columnas:

USER
: Dueño del proceso.

PID
: Identificador de proceso.

%CPU
: Porcentaje de CPU utilizado.

%MEM
: Porcentaje de memoria física utilizado.

VSZ
: Memoria virtual de procesos en KiB.

RSS
: Memoria física no intercambiada utilizada por el proceso en KiB.

TT
: Terminal (`tty`) que controla el proceso.

STAT
: Código que representa el estado del proceso. Además de `S`, `R` y `Z` (que vimos al describir la salida de `top`), otros valores posibles incluyen: `D` (suspensión ininterrumpida — generalmente esperando E/S), `T` (detenido — normalmente por una señal de control). Algunos modificadores adicionales incluyen: `<` (alta prioridad — no agradable para otros procesos), `N` (baja prioridad — agradable para otros procesos) o `+` (en el grupo de procesos en primer plano).

STARTED
: Hora a la que comenzó el proceso.

TIME
: Tiempo de CPU acumulado.

COMMAND
: Comando que inició el proceso.

Ejercicios Guiados

1. `oneko` es un programa divertido y agradable que muestra un gato persiguiendo el cursor del mouse. Si aún no está instalado en su sistema de escritorio, instálelo utilizando el administrador de paquetes de su distribución. Lo usaremos para estudiar el control del trabajo.
   - Inicia el programa. ¿Cómo lo hace?
   - Mueva el cursor del mouse para ver cómo lo persigue el gato. Ahora suspenda el proceso. ¿Cómo hace eso? ¿Cuál es el resultado?
   - Compruebe cuántos trabajos tiene actualmente. ¿Qué escribe? ¿Cuál es el resultado?
   - Ahora envíelo al segundo plano especificando su ID de trabajo. ¿Cuál es el resultado? ¿Cómo puede saber que el trabajo se está ejecutando en segundo plano?
   - Finalmente, finalice el trabajo especificando su ID de trabajo. ¿Qué escribes?
2. Descubra los PID de todos los procesos generados por *Apache HTTPD web server* (`apache2`) con dos comandos diferentes:
3. Termine todos los procesos `apache2` sin usar sus PID y con dos comandos diferentes:
4. Suponga que tiene que terminar todas las instancias de `apache2` y no tiene tiempo para averiguar cuáles son sus PID. ¿Cómo lograría eso usando `kill` con la señal predeterminada `SIGTERM` en una línea:

5. Inicie `top` e interactúe con él realizando lo siguiente:
   - Mostrar una vista del bosque de procesos:

   - Muestra rutas completas de procesos que diferencian entre espacio de usuario y espacio de kernel:

6. Escriba el comando `ps` para mostrar todos los procesos iniciados por *Apache HTTPD web server* usuario (`www-data`):
   - Usando la sintaxis BSD:

   - Usando la sintaxis UNIX:

   - Usando la sintaxis de GNU:

Ejercicios Exploratorios

1. La señal `SIGHUP` puede usarse como una forma de reiniciar ciertos demonios. Con *Apache HTTPD web server*, por ejemplo, enviar `SIGHUP` al proceso padre (el que comenzó con `init`) mata a sus hijos. Sin embargo, el padre vuelve a leer sus archivos de configuración, vuelve a abrir los archivos de registro y genera un nuevo conjunto de hijos. Realice las siguientes tareas:
   - Inicie el servidor web:
   - Asegúrese de conocer el PID del proceso principal:
   - Haga que el servidor web Apache HTTPD se reinicie enviándole la señal `SIGHUP` al proceso principal:
   - Verifique que el padre no haya sido terminado y que se hayan generado nuevos hijos:
2. Aunque inicialmente estático, la salida de `ps` puede *volverse* dinámica combinando `ps` y `watch`. Supervisaremos *Apache HTTPD web server* para detectar nuevas conexiones. Antes de realizar las tareas descritas a continuación, se recomienda que lea la descripción de la directiva `MaxConnectionsPerChild` en Apache MPM Common Directives.
   - Agregue la directiva `MaxConnectionsPerChild` con un valor de `1` en el archivo de configuración de `apache2` — en *Debian* y derivados se encuentran en `/etc/apache2/apache2.conf`; en la familia *CentOS*, en `/etc/httpd/conf/httpd.conf`. No olvide reiniciar `apache2` para que los cambios surtan efecto.
   - Escriba un comando que use `watch`, `ps` y `grep` para las conexiones `apache2`.
   - Ahora abra un navegador web o use un navegador de línea de comandos como `lynx` para establecer una conexión con el servidor web a través de su dirección IP. ¿Qué observa en la salida de `watch`?

3. Como has aprendido, por defecto, `top` clasifica las tareas por porcentaje de uso de CPU en orden descendente (los valores más altos en la parte superior). Este comportamiento se puede modificar con las teclas interactivas `M` (uso de memoria), `N` (identificador único del proceso), `T` (tiempo de ejecución) y `P` (porcentaje del tiempo de CPU). Sin embargo, también puede ordenar la lista de tareas a su gusto iniciando `top` con la opción `-o` (para obtener más información, consulte la página `man` de top). Ahora, realice las siguientes tareas:
   - Inicie `top` para que las tareas se ordenen por uso de memoria:
   - Verifique que envió el comando correcto resaltando la columna de memoria:
4. `ps` también tiene una opción `o` para especificar las columnas que desea que se muestren. Investigue esta opción y realice las siguientes tareas:
   - Inicie `ps` para que solo se muestre información sobre *usuario*, *porcentaje de memoria utilizada*, *porcentaje de tiempo de CPU utilizado* y *comando completo*:
   - Ahora, ejecute `ps` para que la única información que se muestre sea la del usuario y el nombre de los programas que están utilizando:

Resumen

En esta lección has aprendido sobre *jobs* y *control de jobs*. Los hechos y conceptos importantes a tener en cuenta son:

- Los trabajos son procesos que se envían en segundo plano.
- Además de un *ID de proceso*, a los trabajos también se les asigna un *ID de trabajo* cuando se crean.
- Para controlar trabajos, se requiere una especificación de trabajo (`jobspec`).
- Los trabajos se pueden poner en primer plano, enviar a un segundo plano, suspender y finalizar (o *matar*).
- Se puede separar un trabajo del terminal y la sesión en la que se creó.

Asimismo, también hemos discutido el concepto de *procesado* y *monitoreo de procesos*. Las ideas más relevantes son:

- Los procesos ejecutan programas.
- Los procesos pueden ser monitoreados.
- Las diferentes utilidades nos permiten encontrar el *ID de proceso* de los procesos y enviarles señales para finalizarlos.
- Las señales se pueden especificar por nombre (por ejemplo, `-SIGTERM`), número (por ejemplo, `-15`) u opción (por ejemplo, `-s SIGTERM`).
- `top` y `ps` son muy potentes cuando se trata de monitorear procesos. La salida del primero es dinámica y se actualiza constantemente; por otro lado, `ps` revela la salida estáticamente.

Comandos utilizados en esta lección:

`jobs`
: Mostrar trabajos activos y su estado.

`sleep`
: Retraso por una cantidad de tiempo específica.

`fg`
: Mover trabajos a primer plano.

`bg`
: Mover trabajos a un segundo plano.

`kill`
: Terminar trabajos.

`nohup`
: Separar trabajos de la sesión/terminal.

`exit`
: Salir del shell actual.

`tail`
: Mostrar las líneas más recientes en un archivo.

`watch`
: Ejecuta un comando repetidamente (ciclo de 2 segundos por defecto).

`uptime`
: Mostrar cuánto tiempo ha estado funcionando el sistema, la cantidad de usuarios actuales y el promedio de carga del sistema.

`free`
: Mostrar uso de memoria.

`pgrep`
: Buscar la identificación del proceso basada en el nombre.

`pidof`
: Buscar la identificación del proceso basada en el nombre.

`pkill`
: Enviar señal para procesar por nombre.

`killall`
: Eliminar procesos por nombre.

`top`
: Mostrar procesos de Linux.

`ps`

Informar en un momento dado de los procesos actuales.

Respuestas a los ejercicios guiados

1. `oneko` es un programa divertido y agradable que muestra un gato persiguiendo el cursor del mouse. Si aún no está instalado en su sistema de escritorio, instálelo utilizando el administrador de paquetes de su distribución. Lo usaremos para estudiar el control del trabajo.

   - Inicie el programa. ¿Cómo lo hace?

     Teclee `oneko` en la terminal.

   - Mueva el cursor del mouse para ver cómo lo persigue el gato. Ahora suspenda el proceso. ¿Cómo hace eso? ¿Cuál es el resultado?

     Presione `Ctrl`+`z`:

     ```
     [1]+  Stopped                 oneko
     ```

   - Compruebe cuántos trabajos tiene actualmente. ¿Qué escribe? ¿Cuál es el resultado?

     ```
     $ jobs
     [1]+  Stopped                 oneko
     ```

   - Ahora envíelo al fondo especificando su ID de trabajo. ¿Cuál es el resultado? ¿Cómo puede saber que el trabajo se está ejecutando en segundo plano?

     ```
     $ bg %1
     [1]+ oneko &
     ```

     El gato se mueve de nuevo.

   - Finalmente, finalice el trabajo especificando su ID de trabajo. ¿Qué escribe?

     ```
     $ kill %1
     ```

2. Descubra los PID de todos los procesos generados por *Apache HTTPD web server* (`apache2`) con dos comandos diferentes:

   ```
   $ pgrep apache2
   ```

o

```
$ pidof apache2
```

3. Termine todos los procesos `apache2` sin usar sus PID y con dos comandos diferentes:

```
$ pkill apache2
```

o

```
$ killall apache2
```

4. Suponga que tiene que terminar todas las instancias de `apache2` y no tiene tiempo para averiguar cuáles son sus PID. ¿Cómo lograría eso usando `kill` con la señal predeterminada `SIGTERM` en una línea:

```
$ kill $(pgrep apache2)
$ kill `pgrep apache2`
```

o

```
$ kill $(pidof apache2)
$ kill `pidof apache2`
```

NOTE Dado que `SIGTERM` (`15`) es la señal predeterminada, no es necesario pasar ninguna opción para `kill`.

5. Inicie `top` e interactúe con él realizando lo siguiente:
   - Mostrar una vista del bosque de procesos:

     Presione `V`.
   - Muestra rutas completas de procesos que diferencian entre espacio de usuario y espacio de kernel:

     Presione `c`.

6. Escriba el comando `ps` para mostrar todos los procesos iniciados por *Apache HTTPD web server* usuario (`www-data`):
   - Usando la sintaxis BSD:

     ```
     $ ps U www-data
     ```

   - Usando la sintaxis UNIX:

     ```
     $ ps -u www-data
     ```

   - Usando la sintaxis de GNU:

     ```
     $ ps --user www-data
     ```

Respuestas a ejercicios exploratorios

1. La señal `SIGHUP` puede usarse como una forma de reiniciar ciertos demonios. Con el *Apache HTTPD web server*, por ejemplo, enviar `SIGHUP` al proceso padre (el que comenzó con `init`) mata a sus hijos. Sin embargo, el padre vuelve a leer sus archivos de configuración, vuelve a abrir los archivos de registro y genera un nuevo conjunto de hijos. Realice las siguientes tareas:
   - Inicie el servidor web:

     ```
     $ sudo systemctl start apache2
     ```

   - Asegúrese de conocer el PID del proceso principal:

     ```
     $ ps aux | grep apache2
     ```

     El proceso padre es el iniciado por el usuario `root`. En nuestro caso el que tiene PID `1653`.

   - Haga que el servidor web Apache HTTPD se reinicie enviándole la señal `SIGHUP` al proceso principal:

     ```
     $ kill -SIGHUP 1653
     ```

   - Verifique que el padre no haya sido terminado y que se hayan generado nuevos hijos:

     ```
     $ ps aux | grep apache2
     ```

     Ahora debería ver el proceso padre `apache2` junto con dos hijos nuevos.

2. Aunque inicialmente estático, la salida de `ps` puede *volverse* dinámica combinando `ps` y `watch`. Supervisaremos *Apache HTTPD web server* para detectar nuevas conexiones. Antes de realizar las tareas descritas a continuación, se recomienda que lea la descripción de la directiva `MaxConnectionsPerChild` en Apache MPM Common Directives.
   - Agregue la directiva `MaxConnectionsPerChild` con un valor de `1` en el archivo de configuración de `apache2` — en *Debian* y derivados se encuentran en `/etc/apache2/apache2.conf`; en la familia *CentOS*, en `/etc/httpd/conf/httpd.conf`. No olvide reiniciar `apache2` para que los cambios surtan efecto.

     La línea a incluir en el archivo de configuración es `MaxConnectionsPerChild 1`. Una forma

de reiniciar el servidor web es a través de `sudo systemctl restart apache2`.

- Escriba un comando que use `watch`, `ps` y `grep` para las conexiones `apache2`.

```
$ watch 'ps aux | grep apache2'
```

o

```
$ watch "ps aux | grep apache2"
```

- Ahora abra un navegador web o use un navegador de línea de comandos como `lynx` para establecer una conexión con el servidor web a través de su dirección IP. ¿Qué observas en la salida de `watch`?

  Uno de los procesos secundarios propiedad de `www-data` desaparece.

3. Como has aprendido, por defecto, `top` clasifica las tareas por porcentaje de uso de CPU en orden descendente (los valores más altos en la parte superior). Este comportamiento se puede modificar con las teclas interactivas `M` (uso de memoria), `N` (identificador único del proceso), `T` (tiempo de ejecución) y `P` (porcentaje del tiempo de CPU). Sin embargo, también puede ordenar la lista de tareas a su gusto iniciando `top` con la opción `-o` (para obtener más información, consulte la página `man` de top). Ahora, realice las siguientes tareas:

   - Inicie `top` para que las tareas se ordenen por uso de memoria:

```
$ top -o %MEM
```

   - Verifique que escribió el comando correcto resaltando la columna de memoria:

     Presione `x`.

4. `ps` también tiene una opción `o` para especificar las columnas que desea que se muestren. Investigue esta opción y realice las siguientes tareas:

   - Inicie `ps` para que solo se muestre información sobre *usuario*, *porcentaje de memoria utilizada*, *porcentaje de tiempo de CPU utilizado* y *comando completo*:

```
$ ps o user,%mem,%cpu,cmd
```

◦ Ahora, ejecute `ps` para que la única información que se muestre sea la del usuario y el nombre de los programas que están utilizando:

```
$ ps o user,comm
```



103.5 Lección 2

Certificación:	LPIC-1
Versión:	5.0
Tema:	103 Comandos GNU y Unix
Objetivo:	103.5 Crear, monitorear y matar procesos
Lección:	2 de 2

Introducción

Las herramientas y utilidades vistas en la lección anterior son muy útiles para el monitoreo de procesos en general. Sin embargo, un administrador del sistema puede necesitar ir un paso más allá. En esta lección discutiremos el concepto de multiplexores de terminales y aprenderemos sobre *GNU Screen* y *tmux* ya que, a pesar de los modernos y excelentes emuladores de terminales actuales, los multiplexores aún conservan algunas características interesantes y potentes para un administrador de sistemas productivo.

Características de los multiplexores terminales

En electrónica, un multiplexor (o *mux*) es un dispositivo que permite conectar múltiples entradas a una sola salida. Por lo tanto, un multiplexor terminal nos da la capacidad de cambiar entre diferentes entradas según sea necesario. Aunque no son exactamente lo mismo, `screen` y `tmux` comparten una serie de características comunes:

- Cualquier invocación exitosa dará como resultado al menos una sesión que, a su vez, incluirá al menos una ventana. Las ventanas contienen programas.
- Las ventanas se puede dividir en regiones o paneles, lo que puede ayudar a la productividad

cuando se trabaja con varios programas simultáneamente.

- Facilidad de control: para ejecutar la mayoría de los comandos, utilizan una combinación de teclas, el llamado *comando prefijo* o *comando clave*, seguido de otro caracter.
- Las sesiones se pueden separar de su terminal actual (es decir, los programas se envían en segundo plano y continúan ejecutándose). Esto garantiza la ejecución completa de los programas sin importar si cerramos accidentalmente un terminal, experimentamos un congelamiento ocasional del terminal o incluso una pérdida de conexión remota.
- Conexiones de Socket.
- Modo de copia.
- Son altamente personalizables.

GNU Screen

En los primeros días de Unix (1970-1980), las computadoras consistían básicamente en terminales conectados a una computadora central. Eso fue todo, sin múltiples ventanas o pestañas. Y esa fue la razón detrás de la creación de GNU Screen en 1987: emular múltiples pantallas *VT100 independientes* en un solo terminal físico.

Ventanas

La pantalla GNU se invoca simplemente escribiendo `screen` en la terminal. Primero verá un mensaje de bienvenida:

```
GNU Screen version 4.05.00 (GNU) 10-Dec-16

Copyright (c) 2010 Juergen Weigert, Sadrul Habib Chowdhury
Copyright (c) 2008, 2009 Juergen Weigert, Michael Schroeder, Micah Cowan, Sadrul
Habib Chowdhury
Copyright (c) 1993-2002, 2003, 2005, 2006, 2007 Juergen Weigert, Michael Schroeder
Copyright (c) 1987 Oliver Laumann
(...)
```

Presione la barra espaciadora o Enter para cerrar el mensaje y se le mostrará un símbolo del sistema:

```
$
```

Puede parecer que no ha pasado nada, pero el hecho es que `screen` ya ha creado y gestionado su primera sesión y ventana. El prefijo de comando de la pantalla es `Ctrl` + `a`. Para ver todas las ventanas

en la parte inferior de la pantalla del terminal, escriba `Ctrl` + `a` - `w`:

```
0*$ bash
```

¡Ahí está, nuestra única ventana hasta ahora! Sin embargo, tenga en cuenta que el conteo inicia en 0. Para crear otra ventana, escriba `Ctrl` + `a` - `c`. Verá un nuevo mensaje. Comencemos con `ps` en esa nueva ventana:

```
$ ps
  PID TTY          TIME CMD
  974 pts/2    00:00:00 bash
  981 pts/2    00:00:00 ps
```

y escriba `Ctrl` + `a` - `w` nuevamente:

```
0-$ bash  1*$ bash
```

Ahí tenemos nuestras dos ventanas (observe el asterisco que indica la que se está mostrando en este momento). Sin embargo, como comenzaron con Bash, ambas reciben el mismo nombre. Ya que invocamos `ps` en nuestra ventana actual, cambiemos el nombre con el mismo nombre. Para eso, debe escribir `Ctrl` + `a` - `A` y escribir el nombre de la nueva ventana (`ps`) cuando se le solicite:

```
Set window's title to: ps
```

Ahora, creemos otra ventana pero proporcione un nombre desde el principio: `yetanotherwindow`. Esto se hace invocando `screen` con el modificador `-t`:

```
$ screen -t yetanotherwindow
```

Puede moverse entre ventanas de diferentes formas:

- Usando `Ctrl` + `a` - `n` (ir a la ventana *siguiente*) y `Ctrl` + `a` - `p` (ir a la ventana *anterior*).
- Usando `Ctrl` + `a` - `número` (vaya a la ventana número *número*).
- Usando `Ctrl` + `a` - `"` para ver una lista de todas las ventanas. Puede moverse hacia arriba y hacia abajo con las teclas de flecha y seleccionar la que desee con Enter:

```
Num Name                                                    Flags

   0 bash                                                       $
   1 ps                                                         $
   2 yetanotherwindow
```

Al trabajar con ventanas, es importante recordar lo siguiente:

- Las ventanas ejecutan sus programas de forma completamente independiente entre sí.
- Los programas seguirán ejecutándose incluso si su ventana no está visible (también cuando se desconecta la sesión de pantalla, como veremos en breve).

Para eliminar una ventana, simplemente finalice el programa que se está ejecutando en ella (una vez que se elimine la última ventana, `screen` terminará). Alternativamente, use `Ctrl` + `a` - `k` mientras está en la ventana que desea eliminar; se le pedirá confirmación:

```
Really kill this window [y/n]

Window 0 (bash) killed.
```

Regiones

`screen` puede dividir la pantalla de un terminal en varias regiones para acomodar las ventanas. Estas divisiones pueden ser horizontales (`Ctrl` + `a` - `S`) o verticales (`Ctrl` + `a` - `|`).

Lo único que mostrará la nueva región es simplemente `--` en la parte inferior, lo que significa que está vacío:

```
   1 ps                                                                --
```

Para moverse a la nueva región, escriba `Ctrl` + `a` - `Tab`. Ahora puede agregar una ventana mediante cualquiera de los métodos que ya hemos visto, por ejemplo: `Ctrl` + `a` - `2`. Ahora el `--` debería haberse convertido en `2 yetanotherwindow`:

```
$ ps                                                  $
  PID TTY          TIME CMD
 1020 pts/2    00:00:00 bash
 1033 pts/2    00:00:00 ps
$ screen -t yetanotherwindow





   1 ps                                                                2
yetanotherwindow
```

Los aspectos importantes a tener en cuenta al trabajar con regiones son:

- Puede moverse entre regiones escribiendo `Ctrl` + `a` - `Tab`.
- Puede terminar todas las regiones excepto la actual con `Ctrl` + `a` - `Q`.
- Puede terminar la región actual con `Ctrl` + `a` - `X`.
- Terminar una región no termina su ventana asociada.

Sesiones

Hasta ahora hemos jugado con algunas ventanas y regiones, pero todas pertenecen a la misma y única sesión. Es hora de empezar a jugar con sesiones. Para ver una lista de todas las sesiones, teclee `screen -list` o `screen -ls`:

```
$ screen -list
There is a screen on:
        1037.pts-0.debian       (08/24/19 13:53:35)     (Attached)
1 Socket in /run/screen/S-carol.
```

Esa es nuestra única sesión hasta ahora:

PID

`1037`

Name

`pts-0.debian` (indicando el terminal—en nuestro caso un *pseudo terminal esclavo*—y el nombre de host).

Status

`Attached`

Creemos una nueva sesión dándole un nombre más descriptivo:

```
$ screen -S "second session"
```

La pantalla del terminal se borrará y se le dará un nuevo mensaje. Puede comprobar las sesiones una vez más:

```
$ screen -ls
There are screens on:
        1090.second session     (08/24/19 14:38:35)     (Attached)
        1037.pts-0.debian       (08/24/19 13:53:36)     (Attached)
2 Sockets in /run/screen/S-carol.
```

Para cerrar una sesión, salga de todas sus ventanas o simplemente escriba el comando `screen -S SESSION-PID -X quit` (también puede proporcionar el nombre de la sesión). Deshagámonos de nuestra primera sesión:

```
$ screen -S 1037 -X quit
```

Se le enviará de vuelta al indicador de su terminal fuera de la `screen`. Pero recuerde, nuestra segunda sesión todavía está viva:

```
$ screen -ls
There is a screen on:
    1090.second session (08/24/19 14:38:35) (Detached)
1 Socket in /run/screen/S-carol.
```

Sin embargo, dado que matamos su sesión principal, se le asigna una nueva etiqueta: `Detached`.

Desvincular sesiones

Por varias razones, es posible que desee desconectar una sesión de pantalla de su terminal:

- Para permitir que su computadora en el trabajo haga su trabajo y se conecte de forma remota más tarde desde casa.

- Para compartir una sesión con otros usuarios.

Desconecte una sesión con la combinación de teclas Ctrl + a - d. Volverá a su terminal:

```
[detached from 1090.second session]
$
```

Para vincular nuevamente a la sesión, use el comando `screen -r SESSION-PID`. Alternativamente, puede usar el `SESSION-NAME` como vimos arriba. Si solo hay una sesión desvinculada, ninguna es obligatoria:

```
$ screen -r
```

Este comando es suficiente para volver a vincularla a nuestra segunda sesión:

```
$ screen -ls
There is a screen on:
        1090.second session      (08/24/19 14:38:35)     (Attached)
1 Socket in /run/screen/S-carol.
```

Opciones importantes para volver a adjuntar la sesión:

`-d -r`
: Vuelva a conectar una sesión y, si es necesario, desconéctela primero.

`-d -R`
: Igual que `-d -r` pero `screen` incluso creará la sesión primero si no existe.

`-d -RR`
: Igual que `-d -R`. Sin embargo, utilice la primera sesión si hay más de una disponible.

`-D -r`
: Vuelva a conectar una sesión. Si es necesario, desconecte y cierre la sesión de forma remota primero.

`-D -R`
: Si se está ejecutando una sesión, vuelva a conectarla (desconecte y cierre la sesión de forma remota primero si es necesario). Si no se estaba ejecutando créelo y notifique al usuario.

`-D -RR`

Lo mismo que `-D -R` - solo que más fuerte.

`-d -m`

Inicie `screen` en *modo independiente*. Esto crea una nueva sesión pero no se vincula a ella. Esto es útil para los scripts de inicio del sistema.

`-D -m`

Igual que `-d -m`, pero no bifurca un nuevo proceso. El comando sale si la sesión termina.

Lea las páginas del manual de `screen` para conocer otras opciones.

Copiar y pegar: modo Scrollback

GNU Screen presenta un modo de copia o *scrollback*. Una vez ingresado, puede mover el cursor en la ventana actual y por el contenido de su historial usando las teclas de flecha. Puede marcar texto y copiarlo en ventanas. Los pasos a seguir son:

1. Ingrese al modo de copia/scrollback: `Ctrl` + `a` - `[`.
2. Vaya al principio del texto que desea copiar usando las teclas de flecha.
3. Marque el comienzo del fragmento de texto que desea copiar: Espacio.
4. Vaya al final del fragmento de texto que desea copiar usando las teclas de flecha.
5. Marque el final del fragmento de texto que desea copiar: Espacio.
6. Vaya a la ventana de su elección y pegue el fragmento de texto: `Ctrl` + `a` - `]`.

Personalización de Screen

El archivo de configuración de todo el sistema para screen es `/etc/screenrc`. Alternativamente, se puede usar un `~/.screenrc` a nivel de usuario. El archivo incluye cuatro secciones principales de configuración:

`SCREEN SETTINGS`

Puede definir la configuración general especificando la *directiva* seguida de un espacio y el *valor* como en: `defscrollback 1024`.

`SCREEN KEYBINDINGS`

Esta sección es bastante interesante ya que le permite redefinir combinaciones de teclas que quizás interfieran con su uso diario del terminal. Utilice la palabra clave `bind` seguida de un

espacio, el carácter que se utilizará después del prefijo del comando, otro espacio y el comando como en: `bind l kill` (esta configuración cambiará la forma predeterminada de matar una ventana a `Ctrl` + `a` - `l`).

Para mostrar todos los enlaces de la pantalla, escriba `Ctrl` + `a` - `?` O consulte la página del manual.

TIP Por supuesto, también puede cambiar el prefijo del comando. Por ejemplo, para ir de `Ctrl` + `a` a `Ctrl` + `b`, simplemente agregue esta línea: `escape ^Bb`.

`TERMINAL SETTINGS`

Esta sección incluye configuraciones relacionadas con el tamaño de la ventana de la terminal y los búferes, entre otros. Para habilitar el modo sin bloqueo para manejar mejor las conexiones ssh inestables, por ejemplo, se utiliza la siguiente configuración: `defnonblock 5`.

`STARTUP SCREENS`

Puede incluir comandos para que varios programas se ejecuten en el inicio de la `pantalla`; por ejemplo: `screen -t top top` (screen abrirá una ventana llamada `top` con `top` adentro).

tmux

`tmux` fue lanzado en 2007. Aunque es muy similar a `screen`, incluye algunas diferencias notables:

- Modelo cliente-servidor: el servidor suministra una serie de sesiones, cada una de las cuales puede tener varias ventanas vinculadas que, a su vez, pueden ser compartidas por varios clientes.
- Selección interactiva de sesiones, ventanas y clientes a través de menús.
- La misma ventana se puede vincular a varias sesiones.
- Disponibilidad de diseños de teclas *vim* y *Emacs*.
- Soporte para terminales UTF-8 y 256 colores.

Ventanas

`tmux` se puede invocar simplemente escribiendo `tmux` en el símbolo del sistema. Se le mostrará un indicador de shell y una barra de estado en la parte inferior de la ventana:

```
[0] 0:bash*                                                "debian" 18:53
27-Aug-19
```

Aparte del `hostname`, la hora y la fecha, la barra de estado proporciona la siguiente información:

Session name

`[0]`

Window number

`0:`

Window name

`bash*`. Por defecto, este es el nombre del programa que se ejecuta dentro de la ventana y, a diferencia de `screen`, `tmux` lo actualizará automáticamente para reflejar el programa en ejecución actual. Note el asterisco que indica que esta es la ventana visible actual.

Puede asignar nombres de sesión y ventana al invocar `tmux`:

```
$ tmux new -s "LPI" -n "Window zero"
```

La barra de estado cambiará en consecuencia:

```
[LPI] 0:Window zero*                                    "debian" 19:01
27-Aug-19
```

El prefijo de comando de tmux es `Ctrl` + `b`. Para crear una nueva ventana, simplemente escriba `Ctrl` + `b`-`c`; se le llevará a un nuevo prompt y la barra de estado reflejará la nueva ventana:

```
[LPI] 0:Window zero- 1:bash*                            "debian" 19:02
27-Aug-19
```

Como Bash es el shell subyacente, la nueva ventana recibe ese nombre de forma predeterminada. Inicie `top` y vea cómo cambia el nombre a `top`:

```
[LPI] 0:Window zero- 1:top*                            "debian" 19:03
27-Aug-19
```

En cualquier caso, puede cambiar el nombre de una ventana con `Ctrl` + `b`-`,`. Cuando se le solicite, proporcione el nuevo nombre y presione Enter:

```
(rename-window) Window one
```


Puede mostrar todas las ventanas para su selección con `Ctrl` + `b` - `w` (use las teclas de flecha para moverse hacia arriba y hacia abajo y `enter` para seleccionar):

```
(0)  0: Window zero- "debian"
(1)  1: Window one* "debian"
```

De manera similar a `screen`, podemos saltar de una ventana a otra con:

`Ctrl` + `b` - `n`
: ir a la *siguiente* ventana.

`Ctrl` + `b` - `p`
: ir a la ventana *anterior*.

`Ctrl` + `b` - `número`
: ir a la ventana *número*.

Para deshacerse de una ventana, use `Ctrl` + `b` - `&`. Se le pedirá confirmación:

```
kill-window Window one? (y/n)
```

Otros comandos de ventana interesantes incluyen:

`Ctrl` + `b` - `f`
: buscar ventana por nombre.

`Ctrl` + `b` - `.`
: cambiar el número de índice de la ventana.

Para leer la lista completa de comandos, consulte la página del manual.

Paneles

La función de división de ventanas de `screen` también está presente en `tmux`. Sin embargo, las divisiones resultantes no se llaman *regiones* sino *paneles*. La diferencia más importante entre regiones y paneles es que estos últimos son pseudo-terminales completas vinculadas a una ventana. Esto significa que matar un panel también matará su pseudo-terminal y cualquier programa asociado que se ejecute dentro.

Para dividir una ventana horizontalmente, usamos `Ctrl` + `b` - `"`:

```
Tasks:  93 total,   1 running,  92 sleeping,   0 stopped,   0 zombie
%Cpu(s):  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
KiB Mem :  4050960 total,  3730920 free,   114880 used,   205160 buff/cache
KiB Swap:  4192252 total,  4192252 free,        0 used.  3716004 avail Mem

  PID USER      PR  NI    VIRT    RES    SHR S %CPU %MEM     TIME+ COMMAND
 1340 carol     20   0   44876   3400   2800 R  0.3  0.1   0:00.24 top
    1 root      20   0  139088   6988   5264 S  0.0  0.2   0:00.50 systemd
    2 root      20   0       0      0      0 S  0.0  0.0   0:00.00 kthreadd
    3 root      20   0       0      0      0 S  0.0  0.0   0:00.04 ksoftirqd/0
    4 root      20   0       0      0      0 S  0.0  0.0   0:01.62 kworker/0:0
    5 root       0 -20       0      0      0 S  0.0  0.0   0:00.00 kworker/0:0H
    7 root      20   0       0      0      0 S  0.0  0.0   0:00.06 rcu_sched
    8 root      20   0       0      0      0 S  0.0  0.0   0:00.00 rcu_bh
    9 root      rt   0       0      0      0 S  0.0  0.0   0:00.00 migration/0
   10 root       0 -20       0      0      0 S  0.0  0.0   0:00.00 lru-add-drain
   11 root      rt   0       0      0      0 S  0.0  0.0   0:00.01 watchdog/0
   12 root      20   0       0      0      0 S  0.0  0.0   0:00.00 cpuhp/0
$
□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□
□□□□□□□□□□□□
$








[LPI] 0:Window zero- 1:Window one*                                    "debian"
19:05 27-Aug-19
```

Para dividirlo verticalmente, use `Ctrl` + `b` - `%`:

```
                                                                                          ▯ $
    1 root      20   0  139088   6988   5264 S  0.0  0.2   0:00.50 systemd         ▯
                                                                                   ▯
    2 root      20   0       0      0      0 S  0.0  0.0   0:00.00 kthreadd        ▯
                                                                                   ▯
    3 root      20   0       0      0      0 S  0.0  0.0   0:00.04 ksoftirqd/0     ▯
    4 root      20   0       0      0      0 S  0.0  0.0   0:01.62 kworker/0:0     ▯
    5 root       0 -20       0      0      0 S  0.0  0.0   0:00.00 kworker/0:0H▯
    7 root      20   0       0      0      0 S  0.0  0.0   0:00.06 rcu_sched       ▯
    8 root      20   0       0      0      0 S  0.0  0.0   0:00.00 rcu_bh          ▯
    9 root      rt   0       0      0      0 S  0.0  0.0   0:00.00 migration/0     ▯
                                                                                   ▯
   10 root       0 -20       0      0      0 S  0.0  0.0   0:00.00 lru-add-drai▯
n                                                                                  ▯
   11 root      rt   0       0      0      0 S  0.0  0.0   0:00.01 watchdog/0      ▯
                                                                                   ▯
   12 root      20   0       0      0      0 S  0.0  0.0   0:00.00 cpuhp/0         ▯
                                                                                   ▯
$                                                                                  ▯
▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯
▯▯▯▯▯▯▯▯▯▯▯▯
$










[LPI] 0:Window zero- 1:Window one*                                     "debian"
19:05 27-Aug-19
```

Para destruir el panel actual (junto con su pseudo-terminal que se ejecuta dentro de él, junto con cualquier programa asociado), use `Ctrl`+`b`-`x`. Se le pedirá confirmación en la barra de estado:

```
kill-pane 1? (y/n)
```

Comandos de panel importantes:

`Ctrl` + `b` - `↑`, `↓`, `←`, `→`
: moverse entre paneles.

`Ctrl` + `b` - `;`
: pasar al último panel activo.

`Ctrl` + `b` - `Ctrl` + `arrow key`
: cambiar el tamaño del panel en una línea.

`Ctrl` + `b` - `Alt` + `arrow key`
: cambiar el tamaño del panel en cinco líneas.

`Ctrl` + `b` - `{`
: intercambiar paneles (actual a anterior).

`Ctrl` + `b` - `}`
: intercambiar paneles (actual a siguiente).

`Ctrl` + `b` - `z`
: panel de acercar/alejar.

`Ctrl` + `b` - `t`
: `tmux` muestra un reloj elegante dentro del panel (deténgalo presionando `q`).

`Ctrl` + `b` - `!`
: convertir el panel en ventana.

Para leer la lista completa de comandos, consulte la página del manual.

Sesiones

Para listar sesiones en `tmux` puede usar *`Ctrl`* + *`b`* - *`s`*:

```
(0) + LPI: 2 windows (attached)
```

Alternativamente, puede usar el comando `tmux ls`:

```
$ tmux ls
LPI: 2 windows (created Tue Aug 27 19:01:49 2019) [158x39] (attached)
```

Solo hay una sesión (`LPI`) que incluye dos ventanas. Creemos una nueva sesión desde nuestra sesión actual. Esto se puede lograr usando Ctrl + b, teclee `:new` en el indicador, luego presione Enter. Se le enviará a la nueva sesión como se puede observar en la barra de estado:

```
[2] 0:bash*                                                     "debian" 19:15
27-Aug-19
```

Por defecto, `tmux` nombró la sesión `2`. Para cambiarle el nombre, use Ctrl + b - $. Cuando se le solicite, proporcione el nuevo nombre y presione Enter:

```
(rename-session) Second Session
```

Puede cambiar de sesión con Ctrl + b - s (use las teclas de flecha y enter):

```
(0) + LPI: 2 windows
(1) + Second Session: 1 windows (attached)
```

Para cerrar una sesión, puede usar el comando `tmux kill-session -t SESSION-NAME`. Si escribe el comando desde la sesión adjunta actual, se le sacará de `tmux` y volverá a su sesión de terminal inicial:

```
$ tmux kill-session -t "Second Session"
[exited]
$
```

Desvincular sesiones

Al matar a `Second Session` estamos fuera de `tmux`. Sin embargo, todavía tenemos una sesión activa. Pídale a `tmux` una lista de sesiones y seguramente la encontrará allí:

```
$ tmux ls
LPI: 2 windows (created Tue Aug 27 19:01:49 2019) [158x39]
```

Sin embargo, esta sesión está separada de su terminal. Podemos vincularla con `tmux attach -t SESSION-NAME` (`attach` puede ser reemplazado por `at` o — simplemente — `a`). Cuando solo hay una sesión, la especificación del nombre es opcional:

```
$ tmux a
```

Ahora está de vuelta en su sesión; para desplegarla, presione Ctrl + b - d:

```
[detached (from session LPI)]
$
```

TIP La misma sesión se puede vincular a más de un terminal. Si desea vincular una sesión asegurándose de que primero esté desconectada de cualquier otro terminal, use el modificador `-d`: `tmux attach -d -t SESSION-NAME`.

Comandos importantes para vincular/desvincular sesiones:

Ctrl + b - D
: seleccione qué cliente desvincular.

Ctrl + b - r
: Actualizar la terminal del cliente.

Para leer la lista completa de comandos, consulte la página del manual.

Copiar y pegar: modo Scrollback

`tmux` también presenta el modo scrollback básicamente de la misma manera que `screen` (recuerde usar el prefijo de comando de tmux y no el de screen). La única diferencia en cuanto a los comandos es que usa Ctrl+Espacio para marcar el comienzo de la selección y Alt + w para copiar el texto seleccionado.

Personalización de tmux

Los archivos de configuración para `tmux` se encuentran normalmente en `/etc/tmux.conf` y `~/.tmux.conf`. Cuando se inicia, `tmux` utiliza estos archivos si existen. También existe la posibilidad de iniciar `tmux` con el parámetro `-f` para proporcionar un archivo de configuración alternativo. Puede encontrar un ejemplo de archivo de configuración `tmux` ubicado en `/usr/share/doc/tmux/example_tmux.conf`. El nivel de personalización que se puede lograr es

realmente alto. Algunas de las cosas que puede hacer incluyen:

- Cambiar la tecla de prefijo

```
# Change the prefix key to C-a
set -g prefix C-a
unbind C-b
bind C-a send-prefix
```

- Establecer combinaciones de teclas adicionales para ventanas superiores a 9

```
# Some extra key bindings to select higher numbered windows
bind F1 selectw -t:10
bind F2 selectw -t:11
bind F3 selectw -t:12
```

Para obtener una lista completa de todos los enlaces, escriba `Ctrl`+`b`-`?` (Presione `q` para salir) o consulte la página del manual.

Ejercicios Guiados

1. Indique si las siguientes declaraciones/características corresponden a la GNU Screen, tmux o ambos:

Característica/Declaración	GNU Screen	tmux
El prefijo de comando predeterminado es `Ctrl` + `a`		
Modelo cliente-servidor		
Los paneles son pseudo-terminales		
Matar una región no mata sus ventanas asociadas		
Las sesiones incluyen ventanas		
Las sesiones se pueden separar		

2. Instale GNU Screen en su computadora (nombre del paquete: `screen`) y complete las siguientes tareas:

   - Inicie el programa. ¿Qué comando utiliza?

   - Inicie `top`:

   - Usando el prefijo de la tecla de pantalla, abra una nueva ventana; luego, abra `/etc/screenrc` usando `vi`:

   - Liste las ventanas en la parte inferior de la pantalla:

   - Cambie el nombre de la ventana actual a `vi`:

   - Cambie el nombre de la ventana restante a `top`. Para hacer eso, primero muestre una lista de todas las ventanas para que pueda moverse hacia arriba y hacia abajo y seleccionar la

correcta:

◦ Verifique que los nombres hayan cambiado volviendo a mostrar los nombres de las ventanas en la parte inferior de la pantalla:

◦ Ahora, separe la sesión y haga que `screen` cree una nueva llamada `ssh`:

◦ Desconecte también de `ssh` y haga que `screen` muestre la lista de sesiones:

◦ Ahora, adjunte a la primera sesión usando su PID:

◦ Debería estar de vuelta en la ventana que muestra `top`. Divida la ventana horizontalmente y muévase a la nueva región vacía:

◦ Haga que `screen` enumere todas las ventanas y seleccione `vi` para que se muestre en la nueva región vacía:

◦ Ahora, divida la región actual verticalmente, muévase a la región vacía recién creada y asóciela con una nueva ventana:

◦ Termine todas las regiones excepto la actual (recuerde, aunque mate las regiones, las ventanas siguen vivas). Luego, salga de todas las ventanas de la sesión actual hasta que termine la sesión:

◦ Finalmente, haga que `screen` liste sus sesiones una vez más, elimine la sesión `ssh` restante

por PID y verifique que en realidad no quedan sesiones:

3. Instale `tmux` en su computadora (nombre del paquete: `tmux`) y complete las siguientes tareas:
   - Inicie el programa. ¿Qué comando utiliza?
   - Inicie `top` (observe cómo, en un par de segundos, el nombre de la ventana cambia a `top` en la barra de estado):
   - Usando el prefijo de clave de tmux, abra una nueva ventana; luego, cree `~/.tmux.conf` usando `nano`:
   - Divida la ventana verticalmente y reduzca el tamaño del panel recién creado varias veces:
   - Ahora cambie el nombre de la ventana actual a `text editing`; luego, haga que `tmux` muestre una lista con todas sus sesiones:
   - Vaya a la ventana que ejecuta `top` y vuelva a la actual usando la misma combinación de teclas:
   - Desconecte la sesión actual y cree una nueva cuyo nombre sea `ssh` y su nombre de ventana sea `ssh window`:
   - Desconecte también de la sesión `ssh` y haga que `tmux` muestre la lista de sesiones nuevamente:

> **NOTE** A partir de este punto, el ejercicio requiere que utilice una máquina *remota* para las conexiones `ssh` a su host local (una máquina virtual es perfectamente válida y puede resultar realmente práctica). Asegúrese de tener `openssh-server` instalado y ejecutándose en su máquina local y que al menos `openssh-client` esté instalado en la máquina remota.

- Ahora, inicie una máquina remota y conéctese a través de `ssh` con su host local. Una vez que se haya establecido la conexión, verifique las sesiones `tmux`:

- En el host remoto, adjunte a la sesión `ssh` por nombre:

- De vuelta en su máquina local, conéctese a la sesión `ssh` por nombre, asegurándose de que la conexión al host remoto finalice primero:

- Haga que todas las sesiones se muestren para su selección y vaya a su primera sesión (`[0]`). Una vez allí, mate la sesión `ssh` por su nombre:

- Finalmente, desconéctese de la sesión actual y elimínela por su nombre:

Ejercicios Exploratorios

1. Tanto `screen` como `tmux` pueden ingresar al modo de línea de comandos a través de *prefijo de comando* + `:` (ya vimos un breve ejemplo con `tmux`). Investigue un poco y realice las siguientes tareas en el modo de línea de comandos:
   - Haga que `screen` entre en el modo de copia:
   - Haga que `tmux` cambie el nombre de la ventana actual:
   - Haga que `screen` cierre todas las ventanas y finalice la sesión:
   - Haga que `tmux` divida un panel en dos:
   - Haga que `tmux` elimine la ventana actual:

2. Cuando ingresa al modo de copia en `screen`, no solo puede usar las teclas de flecha y `PgUP` o `PgDown` para navegar por la ventana actual y el búfer de retroceso. También existe la posibilidad de utilizar un editor de pantalla completa similar a vi. Con este editor, realice las siguientes tareas:
   - Echo `supercalifragilisticexpialidocious` en su terminal `screen`:
   - Ahora, copie los cinco caracteres consecutivos (de izquierda a derecha) en la línea justo encima de su cursor:
   - Finalmente, pegue la selección (`stice`) en su símbolo del sistema:

3. Suponga que desea compartir una sesión `tmux` (`our_session`) con otro usuario. Ha creado el socket (`/tmp/our_socket`) con los permisos adecuados para que tanto usted como el otro usuario puedan leer y escribir. ¿Qué otras dos condiciones deben cumplirse para que el segundo usuario pueda adjuntar correctamente la sesión a través de `tmux -S /tmp/our_socket a -t our_session`?

Resumen

En esta lección ha aprendido sobre *multiplexores de terminales* en general y GNU Screen y tmux en particular. Los conceptos importantes para recordar incluyen:

- Prefijo de comando: `screen` use Ctrl + a + *caracter*; `tmux`, Ctrl + b + *caracter*.
- Estructura de sesiones, ventanas y divisiones de ventanas (regiones o paneles).
- Modo de copia.
- Separación de sesión: una de las características más potentes de los multiplexores.

Comandos utilizados en esta lección:

`screen`
: Inicia una sesión de pantalla.

`tmux`
: Inicia una sesión `tmux`.

Respuestas a los ejercicios guiados

1. Indique si las siguientes declaraciones/características corresponden a la GNU Screen, tmux o ambos:

Característica/Declaración	GNU Screen	tmux
El prefijo de comando predeterminado es `Ctrl` + `a`	x	
Modelo cliente-servidor		x
Los paneles son pseudo-terminales		x
Matar una región no mata sus ventanas asociadas	x	
Las sesiones incluyen ventanas	x	x
Las sesiones se pueden separar	x	x

2. Instale GNU Screen en su computadora (nombre del paquete: `screen`) y complete las siguientes tareas:
   - Inicie el programa. ¿Qué comando usaría?

     `screen`

   - Inicie `top`:

     `top`

   - Usando el prefijo de la tecla de pantalla, abra una nueva ventana; luego, abra `/etc/screenrc` usando `vi`:

     `Ctrl` + `a` - `c`

     `sudo vi /etc/screenrc`

   - Enumere las ventanas en la parte inferior de la pantalla :

     `Ctrl` + `a` - `w`

   - Cambie el nombre de la ventana actual a `vi`:

     `Ctrl` + `a` - `A`. Escriba `vi` y presione `enter`.

- Cambie el nombre de la ventana restante a `top`. Para hacer eso, primero muestre una lista de todas las ventanas para que pueda moverse hacia arriba y hacia abajo y seleccionar la correcta:

  Primero teclee *Ctrl* + *a* - *"*. Luego usamos las flechas para marcar el que dice `0 bash` y presione *enter*. Finalmente teclee *Ctrl* + *a* - *A*, escriba `top` y presione *enter*.

- Verifique que los nombres hayan cambiado volviendo a mostrar los nombres de las ventanas en la parte inferior de la pantalla:

  Ctrl + *a* - *w*

- Ahora, desvincule la sesión y haga que `screen` cree una nueva llamada `ssh`:

  Ctrl + *a* - *d* `screen -S "ssh"` y presione *enter*.

- Desconecte también de `ssh` y haga que `screen` muestre la lista de sesiones:

  Ctrl + *a* - *d* `screen -list` or `screen -ls`.

- Ahora, vincúlelo a la primera sesión usando su PID:

  `screen -r PID-OF-SESSION`

- Debería estar de vuelta en la ventana que muestra `top`. Divida la ventana horizontalmente y muévase a la nueva región vacía:

  Ctrl + *a* - *S*

  Ctrl + *a* - *Tab*

- Haga que `screen` liste todas las ventanas y seleccione `vi` para que se muestre en la nueva región vacía:

  Usaremos *Ctrl* + *a* - *"* para que se muestren todas las ventanas para su selección, marque `vi` y presione *enter*.

- Ahora, divida la región actual verticalmente, muévase a la región vacía recién creada y asóciela con una nueva ventana:

  Ctrl + *a* - *|*

  Ctrl + *a* - *Tab*

`Ctrl`+`a`-`c`

- Termine todas las regiones excepto la actual (recuerde, aunque mates las regiones, las ventanas siguen vivas). Luego, salga de todas las ventanas de la sesión actual hasta que termine la sesión:

  `Ctrl`+`a`-`Q`. `exit` (para salir de Bash). `Shift`+`:`, posteriormente teclee `quit` y presione `enter` (para salir de `vi`). Después, teclee `exit` (para salir del shell de Bash subyacente) `q` (para terminar `top`); después teclee `exit` (para salir del shell de Bash subyacente).

- Finalmente, haga que `screen` liste sus sesiones una vez más, elimine la sesión `ssh` restante por PID y verifique que en realidad no quedan sesiones:

  `screen -list` o `screen -ls`

  `screen -S PID-OF-SESSION -X quit`

  `screen -list` o `screen -ls`

3. Instale `tmux` en su computadora (nombre del paquete: `tmux`) y complete las siguientes tareas:

   - Inicie el programa. ¿Qué comando usas?

     `tmux`

   - Inicie `top` (observe cómo, en un par de segundos, el nombre de la ventana cambia a `top` en la barra de estado):

     `top`

   - Usando el prefijo de clave de tmux, abra una nueva ventana; luego, crea `~/.tmux.conf` usando `nano`:

     `Ctrl`+`b`-`c` `nano ~/.tmux.conf`

   - Divida la ventana verticalmente y reduzca el tamaño del panel recién creado varias veces:

     `Ctrl`+`b`-`"`

     `Ctrl`+`b`-`Ctrl`+`↓`

   - Ahora cambie el nombre de la ventana actual a `text editing`; luego, haga que `tmux` muestre una lista con todas sus sesiones:

`Ctrl`+`b`-`,`. Luego proporciona el nuevo nombre y presiona `enter`. `Ctrl`+`b`-`s` o `tmux ls`.

- Vaya a la ventana que ejecuta `top` y vuelva a la actual usando la misma combinación de teclas:

  `Ctrl`+`b`-`n` or `Ctrl`+`b`-`p`

- Desconecte de la sesión actual y cree una nueva cuyo nombre sea `ssh` y su nombre de ventana sea `ssh window`:

  `Ctrl`+`b`-`d` `tmux new -s "ssh" -n "ssh window"`

- Desconecte también de la sesión `ssh` y haga que `tmux` muestre la lista de sesiones nuevamente:

  `Ctrl`+`b`-`d` `tmux ls`

NOTE A partir de este punto, el ejercicio requiere que utilice una máquina *remota* para las conexiones `ssh` a su host local (una máquina virtual es perfectamente válida y puede resultar realmente práctica). Asegúrese de tener `openssh-server` instalado y ejecutándose en su máquina local y que al menos `openssh-client` esté instalado en la máquina remota.

- Ahora, inicie una máquina remota y conéctese a través de `ssh` desde su host local. Una vez que se haya establecido la conexión, verifique las sesiones `tmux`:

  En un host remoto: `ssh` *`local-username@local-ipaddress`*. Una vez que este conectado al equipo local: `tmux ls`.

- En el host remoto, adjunte a la sesión `ssh` por nombre:

  `tmux a -t ssh` (`a` puede reemplazarse con `at` o `attach`).

- De vuelta en su máquina local, conéctese a la sesión `ssh` por nombre, asegurándose de que la conexión al host remoto finalice primero:

  `tmux a -d -t ssh` (`a` puede ser reemplazarse por `at` o `attach`).

- Haga que todas las sesiones se muestren para su selección y vaya a su primera sesión (`[0]`). Una vez allí, mata la sesión `ssh` por su nombre:

  Teclee `Ctrl`+`b`-`s`, use las teclas de flecha para marcar la sesión `0` y presione `enter` `tmux kill-`

`session -t ssh`.

- Finalmente, desconéctese de la sesión actual y elimínela por su nombre:

  Ctrl+*b*-*d* `tmux kill-session -t 0`.

Respuestas a ejercicios exploratorios

1. Tanto `screen` como `tmux` pueden ingresar al modo de línea de comandos a través de *prefijo de comando* + `:` (ya vimos un breve ejemplo con `tmux`). Investigue un poco y realice las siguientes tareas en el modo de línea de comandos:
   - Haga que `screen` entre en el modo de copia:

     `Ctrl` + `a` - `:` — después, teclee `copy`.
   - Haga que `tmux` cambie el nombre de la ventana actual:

     `Ctrl` + `b` - `:` — después, teclee `rename-window`.
   - Haga que `screen` cierre todas las ventanas y finalice la sesión:

     `Ctrl` + `a` - `:` — después, teclee `quit`.
   - Haga que `tmux` divida un panel en dos:

     `Ctrl` + `b` - `:` — después, teclee `split-window`.
   - Haga que `tmux` elimine la ventana actual:

     `Ctrl` + `b` - `:` — después, teclee `kill-window`.

2. Cuando ingresa al modo de copia en `screen`, no solo puede usar las teclas de flecha y `PgUP` o `PgDown` para navegar por la ventana actual y el búfer de retroceso. También existe la posibilidad de utilizar un editor de pantalla completa similar a vi. Con este editor, realice las siguientes tareas:
   - Echo `supercalifragilisticexpialidocious` en su terminal `screen`:

     ```
     echo supercalifragilisticexpialidocious
     ```
   - Ahora, copie los cinco caracteres consecutivos (de izquierda a derecha) en la línea justo encima de su cursor:

     Ingresamos al modo de copia: `Ctrl` + `a` - `[` o `Ctrl` + `a` - `:` y luego escribimos `copy`. Luego, nos movemos a la línea de arriba usando `k` y presionamos `espacio` para marcar el comienzo de la selección. Finalmente, avanzamos cuatro caracteres usando `l` y presionamos `espacio` nuevamente para marcar el final de la selección.
   - Finalmente, pegue la selección (`stice`) en su símbolo del sistema:

Ctrl + a -]

3. Suponga que desea compartir una sesión `tmux` (`our_session`) con otro usuario. Ha creado el socket (`/tmp/our_socket`) con los permisos adecuados para que tanto usted como el otro usuario puedan leer y escribir. ¿Qué otras dos condiciones deben cumplirse para que el segundo usuario pueda adjuntar correctamente la sesión a través de `tmux -S /tmp/our_socket a -t our_session`?

   Ambos usuarios deben tener un grupo en común, p. Ej. `multiplexor`. Luego, también debemos cambiar el conector a ese grupo: `chgrp multiplexer /tmp/our_socket`.



103.6 Modificar la prioridad de ejecución de los procesos

Referencia al objetivo del LPI

LPIC-1 v5, Exam 101, Objective 103.6

Importancia

2

Áreas de conocimiento clave

- Conocer la prioridad predeterminada con la que se crea un proceso.
- Ejecutar un programa con una prioridad mayor o menor de la que tiene de forma predeterminada.
- Cambiar la prioridad de un proceso en ejecución.

Lista parcial de archivos, términos y utilidades

- `nice`
- `ps`
- `renice`
- `top`



103.6 Lección 1

Certificación:	LPIC-1
Versión:	5.0
Tema:	103 Comandos GNU y Unix
Objetivo:	103.6 Modificar las prioridades de ejecución de procesos
Lección:	1 de 1

Introducción

Los sistemas operativos capaces de ejecutar más de un proceso al mismo tiempo se denominan sistemas multitarea o multiprocesamiento. Si bien la verdadera simultaneidad solo ocurre cuando hay más de una unidad de procesamiento disponible, incluso los sistemas de un solo procesador pueden imitar la simultaneidad al cambiar entre procesos muy rápidamente. Esta técnica también se emplea en sistemas con muchas CPU equivalentes, o sistemas *multiprocesador simétrico (SMP)*, dado que el número de procesos concurrentes potenciales supera con creces el número de unidades procesadoras disponibles.

De hecho, solo un proceso a la vez puede controlar la CPU. Sin embargo, la mayoría de las actividades del proceso son *llamadas al sistema*, es decir, el proceso en ejecución transfiere el control de la CPU al proceso de un sistema operativo para que realice la operación solicitada. Las llamadas al sistema están a cargo de toda la comunicación entre dispositivos, como asignaciones de memoria, lectura y escritura en sistemas de archivos, impresión de texto en la pantalla, interacción del usuario, transferencias de red, etc. La transferencia del control de la CPU durante una llamada al sistema permite, que el sistema operativo decida si devolver el control de la CPU al proceso anterior o pasarlo a otro proceso. Como las CPU modernas pueden ejecutar instrucciones mucho más rápido de

lo que la mayoría del hardware externo puede comunicarse entre sí, un nuevo proceso de control puede hacer mucho trabajo de la CPU mientras que las respuestas de hardware solicitadas anteriormente aún no están disponibles. Para garantizar el máximo aprovechamiento de la CPU, los sistemas operativos de multiprocesamiento mantienen una cola dinámica de procesos activos en espera de un intervalo de tiempo de la CPU.

Aunque permiten mejorar significativamente la utilización del tiempo de la CPU, depender únicamente de las llamadas al sistema para cambiar entre procesos no es suficiente para lograr un rendimiento multitarea satisfactorio. Un proceso que no realiza llamadas al sistema podría controlar la CPU de forma indefinida. Esta es la razón por la que los sistemas operativos modernos también son *preferentes*, es decir, un proceso en ejecución se puede volver a poner en la cola para que un proceso más importante pueda controlar la CPU, incluso si el proceso en ejecución no ha realizado una llamada al sistema.

El planificador de Linux

Linux, como sistema operativo de multiprocesamiento preventivo, implementa un planificador que organiza la cola de procesos. Más precisamente, el planificador también decide qué subproceso en cola se ejecutará (un proceso puede ramificar muchos subprocesos independientes), pero proceso y subproceso son términos intercambiables en este contexto. Cada proceso tiene dos predicados que intervienen en su programación: la *política de programación* y la *prioridad de programación*.

Hay dos tipos principales de políticas de programación: *políticas en tiempo real* y *políticas normales*. Los procesos bajo una política de tiempo real se programan directamente por sus valores de prioridad. Si un proceso más importante está listo para ejecutarse, se sustituye por un proceso en ejecución menos importante y el proceso de mayor prioridad toma el control de la CPU. Un proceso de menor prioridad obtendrá el control de la CPU solo si los procesos de mayor prioridad están inactivos o esperando la respuesta del hardware.

Cualquier proceso en tiempo real tiene mayor prioridad que un proceso normal. Como sistema operativo de propósito general, Linux ejecuta solo algunos procesos en tiempo real. La mayoría de los procesos, incluidos los programas de usuario y del sistema, se ejecutan según las políticas de programación normales. Los procesos normales suelen tener el mismo valor de prioridad, pero las políticas normales pueden definir reglas de prioridad de ejecución utilizando otro predicado de proceso: el *valor nice*. Para evitar confusiones con las prioridades dinámicas derivadas de valores agradables, las prioridades de programación generalmente se denominan prioridades de programación *estáticas*.

El planificador de Linux se puede configurar de muchas formas diferentes y existen formas aún más complejas de establecer prioridades, pero estos conceptos generales siempre se aplican. Al inspeccionar y ajustar la programación del proceso, es importante tener en cuenta que solo se verán

afectados los procesos bajo la política de programación normal.

Prioridades de lectura

Linux reserva prioridades estáticas que van de 0 a 99 para procesos en tiempo real y los procesos normales se asignan a prioridades estáticas que van de 100 a 139, lo que significa que hay 39 niveles de prioridad diferentes para procesos normales. Los valores más bajos significan una prioridad más alta. La prioridad estática de un proceso activo se puede encontrar en el archivo `sched`, ubicado en su directorio respectivo dentro del sistema de archivos `/proc`:

```
$ grep ^prio /proc/1/sched
prio                   :       120
```

Como se muestra en el ejemplo, la línea que comienza con `prio` da el valor de prioridad del proceso (el proceso PID 1 es el proceso *init* o *systemd*, el primer proceso que el kernel inicia durante la inicialización del sistema). La prioridad estándar para los procesos normales es 120, de modo que se puede reducir a 100 o aumentar a 139. Las prioridades de todos los procesos en ejecución se pueden verificar con el comando `ps -Al` o `ps -el`:

```
$ ps -el
F S   UID   PID  PPID  C PRI  NI ADDR SZ WCHAN  TTY          TIME CMD
4 S     0     1     0  0  80   0 -   9292 -      ?        00:00:00 systemd
4 S     0    19     1  0  80   0 -   8817 -      ?        00:00:00 systemd-journal
4 S   104    61     1  0  80   0 - 64097 -      ?        00:00:00 rsyslogd
4 S     0    63     1  0  80   0 -   7244 -      ?        00:00:00 cron
1 S     0   126     1  0  80   0 -   4031 -      ?        00:00:00 dhclient
4 S     0   154     1  0  80   0 -   3937 -      pts/0    00:00:00 agetty
4 S     0   155     1  0  80   0 -   3937 -      pts/1    00:00:00 agetty
4 S     0   156     1  0  80   0 -   3937 -      pts/2    00:00:00 agetty
4 S     0   157     1  0  80   0 -   3937 -      pts/3    00:00:00 agetty
4 S     0   158     1  0  80   0 -   3937 -      console  00:00:00 agetty
4 S     0   160     1  0  80   0 - 16377 -      ?        00:00:00 sshd
4 S     0   280     0  0  80   0 -   5301 -      ?        00:00:00 bash
0 R     0   392   280  0  80   0 -   7221 -      ?        00:00:00 ps
```

La columna `PRI` indica la prioridad estática asignada por el kernel. Sin embargo, tenga en cuenta que el valor de prioridad mostrado por `ps` difiere del obtenido en el ejemplo anterior. Debido a razones históricas, las prioridades mostradas por `ps` varían de -40 a 99 por defecto, por lo que la prioridad real se obtiene agregando 40 (en particular, 80 + 40 = 120).

También es posible monitorear continuamente los procesos que actualmente administra el kernel de

Linux con el programa `top`. Al igual que con `ps`, `top` también muestra el valor de prioridad de forma diferente. Para que sea más fácil identificar los procesos en tiempo real, `top` resta 100 del valor de prioridad, por lo que todas las prioridades en tiempo real son negativas, con un número negativo o *rt* identificándolas. Por lo tanto, las prioridades normales mostradas por `top` van de 0 a 39.

NOTE

Para obtener más detalles del comando `ps`, se pueden usar opciones adicionales. Compare la salida de este comando con la de nuestro ejemplo anterior:

```
$ ps -e -o user,uid,comm,tty,pid,ppid,pri,pmem,pcpu --sort=-pcpu |
head
```

Niceness de Procesos

Todo proceso normal comienza con un valor *nice* predeterminado de 0 (prioridad 120). El nombre *nice* proviene de la idea de que los procesos "más agradables" permiten que otros procesos se ejecuten antes que ellos en una cola de ejecución particular. Los números *nice* van desde -20 (menos agradables, alta prioridad) a 19 (más agradables, baja prioridad). Linux también permite la capacidad de asignar diferentes valores *nice* a subprocesos dentro del mismo proceso. La columna `NI` en la salida de `ps` indica el número *nice*.

Solo el usuario root puede disminuir el valor *nice* de un proceso por debajo de cero. Es posible iniciar un proceso con una prioridad no estándar con el comando `nice`. Por defecto, `nice` cambia el valor a 10, pero se puede especificar con la opción `-n`:

```
$ nice -n 15 tar czf home_backup.tar.gz /home
```

En este ejemplo, el comando `tar` se ejecuta con un valor *nice* de 15. El comando `renice` puede usarse para cambiar la prioridad de un proceso en ejecución. La opción `-p` indica el número PID del proceso objetivo. Por ejemplo:

```
# renice -10 -p 2164
2164 (process ID) old priority 0, new priority -10
```

Las opciones `-g` y `-u` se utilizan para modificar todos los procesos de un grupo o usuario específico, respectivamente. Con `renice +5 -g users`, el valor *nice* de los procesos propiedad de los usuarios del grupo *users* se elevará en cinco.

Además de `renice`, la prioridad de los procesos se puede modificar con otros programas, como el

programa `top`. En la pantalla principal superior, el valor *nice* de un proceso se puede modificar presionando `r` y luego el número PID del proceso:

```
top - 11:55:21 up 23:38,  1 user,  load average: 0,10, 0,04, 0,05
Tasks:  20 total,   1 running,  19 sleeping,   0 stopped,   0 zombie
%Cpu(s):  0,5 us,  0,3 sy,  0,0 ni, 99,0 id,  0,0 wa,  0,2 hi,  0,0 si,  0,0 st
KiB Mem :  4035808 total,   774700 free,  1612600 used,  1648508 buff/cache
KiB Swap:  7999828 total,  7738780 free,   261048 used.  2006688 avail Mem
PID to renice [default pid = 1]
  PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM     TIME+ COMMAND
    1 root      20   0   74232   7904   6416 S 0,000 0,196   0:00.12 systemd
   15 root      20   0   67436   6144   5568 S 0,000 0,152   0:00.03 systemd-
journal
   21 root      20   0   61552   5628   5000 S 0,000 0,139   0:00.01 systemd-logind
   22 message+  20   0   43540   4072   3620 S 0,000 0,101   0:00.03 dbus-daemon
   23 root      20   0   45652   6204   4992 S 0,000 0,154   0:00.06 wickedd-dhcp4
   24 root      20   0   45648   6276   5068 S 0,000 0,156   0:00.06 wickedd-auto4
   25 root      20   0   45648   6272   5060 S 0,000 0,155   0:00.06 wickedd-dhcp6
```

Aparece el mensaje `PID to renice [default pid = 1]` con el primer proceso listado seleccionado por defecto. Para cambiar la prioridad de otro proceso, escriba su PID y presione Enter. Luego, aparecerá el mensaje `Renice PID 1 to value` (con el número PID solicitado) y se puede asignar un nuevo valor *nice*.

Ejercicios Guiados

1. En un sistema multitarea preventivo, ¿qué sucede cuando un proceso de menor prioridad ocupa el procesador y un proceso de mayor prioridad se pone en cola para su ejecución?

2. Considere la siguiente pantalla `top`:

```
top - 08:43:14 up 23 days, 12:29,  5 users,  load average: 0,13, 0,18, 0,21
Tasks: 240 total,   2 running, 238 sleeping,   0 stopped,   0 zombie
%Cpu(s):  1,4 us,  0,4 sy,  0,0 ni, 98,1 id,  0,0 wa,  0,0 hi,  0,0 si,  0,0 st
MiB Mem :   7726,4 total,    590,9 free,   1600,8 used,   5534,7 buff/cache
MiB Swap:  30517,0 total,  30462,5 free,     54,5 used.   5769,4 avail Mem

  PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM     TIME+ COMMAND
    1 root      20   0  171420  10668   7612 S   0,0   0,1   9:59.15 systemd
    2 root      20   0       0      0      0 S   0,0   0,0   0:02.76 kthreadd
    3 root       0 -20       0      0      0 I   0,0   0,0   0:00.00 rcu_gp
    4 root       0 -20       0      0      0 I   0,0   0,0   0:00.00 rcu_par_gp
    8 root       0 -20       0      0      0 I   0,0   0,0   0:00.00
mm_percpu_wq
    9 root      20   0       0      0      0 S   0,0   0,0   0:49.06
ksoftirqd/0
   10 root      20   0       0      0      0 I   0,0   0,0  18:24.20 rcu_sched
   11 root      20   0       0      0      0 I   0,0   0,0   0:00.00 rcu_bh
   12 root      rt   0       0      0      0 S   0,0   0,0   0:08.17
migration/0
   14 root      20   0       0      0      0 S   0,0   0,0   0:00.00 cpuhp/0
   15 root      20   0       0      0      0 S   0,0   0,0   0:00.00 cpuhp/1
   16 root      rt   0       0      0      0 S   0,0   0,0   0:11.79
migration/1
   17 root      20   0       0      0      0 S   0,0   0,0   0:26.01
ksoftirqd/1
```

¿Qué PID tienen prioridades en tiempo real?

3. Considere la siguiente lista `ps -el`:

```
F S   UID   PID  PPID  C PRI  NI ADDR SZ WCHAN  TTY          TIME CMD
4 S     0     1     0  0  80   0 - 42855 -      ?        00:09:59 systemd
1 S     0     2     0  0  80   0 -     0 -      ?        00:00:02 kthreadd
1 I     0     3     2  0  60 -20 -     0 -      ?        00:00:00 rcu_gp
1 S     0     9     2  0  80   0 -     0 -      ?        00:00:49 ksoftirqd/0
1 I     0    10     2  0  80   0 -     0 -      ?        00:18:26 rcu_sched
1 I     0    11     2  0  80   0 -     0 -      ?        00:00:00 rcu_bh
1 S     0    12     2  0 -40   - -     0 -      ?        00:00:08 migration/0
1 S     0    14     2  0  80   0 -     0 -      ?        00:00:00 cpuhp/0
5 S     0    15     2  0  80   0 -     0 -      ?        00:00:00 cpuhp/1
```

¿Qué PID tiene mayor prioridad?

4. Después de intentar modificar un proceso con `renice`, ocurre el siguiente error:

```
$ renice -10 21704
renice: failed to set priority for 21704 (process ID): Permission denied
```

¿Cuál es la causa probable del error?

Ejercicios Exploratorios

1. Por lo general, es necesario cambiar las prioridades del proceso cuando un proceso ocupa demasiado tiempo de CPU. Usando `ps` con opciones estándar para imprimir todos los procesos del sistema en formato largo, ¿qué valor de `--sort` ordenará los procesos por uso de CPU, aumentando el orden?

2. El comando `schedtool` puede establecer todos los parámetros de programación de la CPU de los que Linux es capaz o mostrar información para procesos dados. ¿Cómo se puede utilizar para mostrar los parámetros de programación del proceso `1750`? Además, ¿cómo se puede usar `schedtool` para cambiar el proceso 1750 a tiempo real con prioridad -90 (como se muestra en `top`)?

Resumen

Esta lección cubre cómo Linux comparte el tiempo de CPU entre sus procesos administrados. Para garantizar el mejor rendimiento, los procesos más críticos deben superar a los procesos menos críticos. La lección pasa por los siguientes pasos:

- Conceptos básicos sobre sistemas multiprocesamiento.
- ¿Qué es un planificador de procesos y cómo lo implementa Linux?
- ¿Cuáles son las prioridades de Linux, números *nice* y su propósito?
- Cómo leer e interpretar las prioridades de los procesos en Linux.
- Cómo cambiar la prioridad de un proceso, antes y durante su ejecución.

Respuestas a los ejercicios guiados

1. En un sistema multitarea preventivo, ¿qué sucede cuando un proceso de menor prioridad ocupa el procesador y un proceso de mayor prioridad se pone en cola para su ejecución?

   El proceso de menor prioridad se detiene y en su lugar se ejecuta el proceso de mayor prioridad.

2. Considere la siguiente pantalla `top`:

```
top - 08:43:14 up 23 days, 12:29,  5 users,  load average: 0,13, 0,18, 0,21
Tasks: 240 total,   2 running, 238 sleeping,   0 stopped,   0 zombie
%Cpu(s):  1,4 us,  0,4 sy,  0,0 ni, 98,1 id,  0,0 wa,  0,0 hi,  0,0 si,  0,0 st
MiB Mem :   7726,4 total,    590,9 free,   1600,8 used,   5534,7 buff/cache
MiB Swap:  30517,0 total,  30462,5 free,     54,5 used.   5769,4 avail Mem

  PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM     TIME+ COMMAND
    1 root      20   0  171420  10668   7612 S   0,0   0,1   9:59.15 systemd
    2 root      20   0       0      0      0 S   0,0   0,0   0:02.76 kthreadd
    3 root       0 -20       0      0      0 I   0,0   0,0   0:00.00 rcu_gp
    4 root       0 -20       0      0      0 I   0,0   0,0   0:00.00 rcu_par_gp
    8 root       0 -20       0      0      0 I   0,0   0,0   0:00.00
mm_percpu_wq
    9 root      20   0       0      0      0 S   0,0   0,0   0:49.06
ksoftirqd/0
   10 root      20   0       0      0      0 I   0,0   0,0  18:24.20 rcu_sched
   11 root      20   0       0      0      0 I   0,0   0,0   0:00.00 rcu_bh
   12 root      rt   0       0      0      0 S   0,0   0,0   0:08.17
migration/0
   14 root      20   0       0      0      0 S   0,0   0,0   0:00.00 cpuhp/0
   15 root      20   0       0      0      0 S   0,0   0,0   0:00.00 cpuhp/1
   16 root      rt   0       0      0      0 S   0,0   0,0   0:11.79
migration/1
   17 root      20   0       0      0      0 S   0,0   0,0   0:26.01
ksoftirqd/1
```

   ¿Qué PIDs tienen prioridades en tiempo real?

   PIDs 12 y 16.

3. Considere la siguiente lista `ps -el`:

```
F S   UID   PID  PPID  C PRI  NI ADDR SZ WCHAN  TTY          TIME CMD
4 S     0     1     0  0  80   0 - 42855 -      ?        00:09:59 systemd
1 S     0     2     0  0  80   0 -     0 -      ?        00:00:02 kthreadd
1 I     0     3     2  0  60 -20 -     0 -      ?        00:00:00 rcu_gp
1 S     0     9     2  0  80   0 -     0 -      ?        00:00:49 ksoftirqd/0
1 I     0    10     2  0  80   0 -     0 -      ?        00:18:26 rcu_sched
1 I     0    11     2  0  80   0 -     0 -      ?        00:00:00 rcu_bh
1 S     0    12     2  0 -40   - -     0 -      ?        00:00:08 migration/0
1 S     0    14     2  0  80   0 -     0 -      ?        00:00:00 cpuhp/0
5 S     0    15     2  0  80   0 -     0 -      ?        00:00:00 cpuhp/1
```

¿Qué PID tiene mayor prioridad?

PID 12.

4. Después de intentar modificar un proceso con `renice`, ocurre el siguiente error:

```
$ renice -10 21704
renice: failed to set priority for 21704 (process ID): Permission denied
```

¿Cuál es la causa probable del error?

Solo el usuario root puede disminuir los números *nice* por debajo de cero.

Respuestas a ejercicios exploratorios

1. Por lo general, es necesario cambiar las prioridades del proceso cuando un proceso ocupa demasiado tiempo de CPU. Usando `ps` con opciones estándar para imprimir todos los procesos del sistema en formato largo, ¿qué valor de `--sort` ordenará los procesos por uso de CPU, aumentando el orden?

```
$ ps -el --sort=pcpu
```

2. El comando `schedtool` puede establecer todos los parámetros de programación de la CPU de los que Linux es capaz o mostrar información para procesos dados. ¿Cómo se puede utilizar para mostrar los parámetros de programación del proceso `1750`? Además, ¿cómo se puede usar `schedtool` para cambiar el proceso 1750 a tiempo real con prioridad -90 (como se muestra en `top`)?

```
$ schedtool 1750
```

```
$ schedtool -R -p 89 1750
```



103.7 Realizar búsquedas en archivos de texto usando expresiones regulares

Referencia al objetivo del LPI

LPIC-1 v5, Exam 101, Objective 103.7

Importancia

3

Áreas de conocimiento clave

- Crear expresiones regulares sencillas que contengan varios elementos de notación.
- Saber diferenciar las expresiones regulares básicas de las extendidas.
- Entender los conceptos de caracteres especiales, clases de caracteres, cuantificadores y anclas.
- Usar herramientas para realizar búsquedas con expresiones regulares dentro de un sistema de archivos o del contenido de un archivo.
- Usar las expresiones regulares para borrar, modificar o reemplazar texto.

Lista parcial de archivos, términos y utilidades

- `grep`
- `egrep`
- `fgrep`
- `sed`
- `regex(7)`



103.7 Lección 1

Certificación:	LPIC-1
Versión:	5.0
Tema:	103 Comandos GNU y Unix
Objetivo:	103.7 Buscar archivos de texto usando expresiones regulares
Lección:	1 de 2

Introducción

Los algoritmos de búsqueda de cadenas son ampliamente utilizados por varias tareas de procesamiento de datos, tanto que los sistemas operativos similares a Unix tienen su propia implementación ubicua: *Expresiones regulares (Regular expressions)*, a menudo abreviadas como *REs*. Las expresiones regulares consisten en secuencias de caracteres que forman un patrón genérico que se utiliza para localizar y, a veces, modificar una secuencia correspondiente en una cadena de caracteres más grande. Las expresiones regulares amplían enormemente la capacidad de:

- Escribir reglas de análisis para solicitudes en servidores HTTP, *nginx* en particular.
- Escribir scripts que conviertan conjuntos de datos basados en texto a otro formato.
- Buscar ocurrencias de interés en entradas de diario o documentos.
- Filtrar documentos de marcado, manteniendo el contenido semántico.

La expresión regular más simple contiene al menos un *átomo*. Un átomo, llamado así porque es el elemento básico de una expresión regular, es sólo un carácter que puede tener o no un significado especial. La mayoría de los caracteres ordinarios son inequívocos, conservan su significado literal,

mientras que otros tienen un significado especial:

`.` (punto)
: Átomo coincide con cualquier carácter.

`^` (signo de intercalación)
: Átomo coincide con el comienzo de una línea.

`$` (signo de dólar)
: Átomo coincide con el final de una línea.

Por ejemplo, la expresión regular `bc`, compuesta por los átomos literales `b` y `c`, se puede encontrar en la cadena `abcd`, pero no en la cadena `a1cd`. Por otro lado, la expresión regular `.c` puede encontrarse en ambas cadenas de caracteres `abcd` y `a1cd`, ya que el punto `.` coincide con cualquier carácter.

Los átomos de signo de intercalación y dólar se utilizan cuando sólo son de interés las coincidencias al principio o al final de la cadena. Por eso también se les llama *anchors* (anclas). Por ejemplo, `cd` se puede encontrar en `abcd`, pero `^cd` no. De manera similar, `ab` se puede encontrar en `abcd`, pero `ab$` no. El signo de intercalación `^` es un carácter literal excepto cuando está al principio y `$` es un carácter literal excepto cuando está al final de la expresión regular.

Expresión de corchetes

Hay otro tipo de átomo llamado *bracket expression (expresión de corchetes)*. Aunque no es un solo carácter, los corchetes `[]` (incluido su contenido) se consideran un solo átomo. Una expresión entre corchetes suele ser sólo una lista de caracteres literales encerrados por `[]`, haciendo que el átomo coincida con cualquier carácter de la lista. Por ejemplo, la expresión `[1b]` se puede encontrar en ambas cadenas, `abcd` y `a1cd`. Para especificar los caracteres a los que no debe corresponder el átomo, la lista debe comenzar con `^`, como en `[^1b]`. También es posible especificar rangos de caracteres en expresiones entre corchetes. Por ejemplo, `[0-9]` coincide con los dígitos del 0 al 9 y `[a-z]` coincide con cualquier letra minúscula. Los rangos deben usarse con precaución, ya que pueden no ser consistentes en distintas configuraciones regionales.

Las listas de expresiones entre corchetes también aceptan clases en lugar de sólo caracteres y rangos individuales. Las clases de caracteres tradicionales son:

`[:alnum:]`
: Representa un carácter alfanumérico.

`[:alpha:]`
: Representa un carácter alfabético.

`[:ascii:]`
: Representa un carácter que encaja en el juego de caracteres ASCII.

`[:blank:]`
: Representa un carácter en blanco, es decir, un espacio o una tabulación.

`[:cntrl:]`
: Representa un carácter de control.

`[:digit:]`
: Representa un dígito (0 a 9).

`[:graph:]`
: Representa cualquier carácter imprimible excepto el espacio.

`[:lower:]`
: Representa un carácter en minúscula.

`[:print:]`
: Representa cualquier carácter imprimible, incluido el espacio.

`[:punct:]`
: Representa cualquier carácter imprimible que no sea un espacio ni un carácter alfanumérico.

`[:space:]`
: Representa caracteres de espacio en blanco: espacio, avance de formulario (`\f`), nueva línea (`\n`), retorno de carro (`\r`), tabulación horizontal (`\t`) y tabulación vertical (`\v`).

`[:upper:]`
: Representa una letra mayúscula.

`[:xdigit:]`
: Representa dígitos hexadecimales (de 0 a F).

Las clases de caracteres se pueden combinar con caracteres y rangos individuales, pero no se pueden usar como punto final de un rango. Además, las clases de caracteres pueden usarse sólo en

expresiones de corchetes, no como un átomo independiente fuera de los corchetes.

Cuantificadores

El alcance de un átomo, ya sea un átomo de un solo carácter o un átomo de corchete, se puede ajustar utilizando un *cuantificador de átomos*. Los cuantificadores de átomos definen *secuencias* de átomos, es decir, las coincidencias ocurren cuando se encuentra una repetición contigua para el átomo en la cadena. La subcadena correspondiente a la coincidencia se llama *pieza*. No obstante, los cuantificadores y otras características de las expresiones regulares se tratan de manera diferente según el estándar que se utilice.

Según lo define POSIX, hay dos tipos de expresiones regulares: expresiones regulares "básicas" y expresiones regulares "extendidas". La mayoría de los programas relacionados con texto en cualquier distribución convencional de Linux admiten ambas formas, por lo que es importante conocer sus diferencias para evitar problemas de compatibilidad y elegir la implementación más adecuada para la tarea prevista.

El cuantificador `*` tiene la misma función tanto en los RE básicos como en los extendidos (el átomo aparece cero o más veces) y es un carácter literal si aparece al principio de la expresión regular o si está precedido por una barra invertida `\`. El cuantificador de signo más `+` seleccionará piezas que contengan una o más coincidencias de átomos en secuencia. Con el cuantificador de signo de interrogación `?`, Se producirá una coincidencia si el átomo correspondiente aparece una vez o si no aparece en absoluto. Si está precedido por una barra invertida `\`, se obvia su significado especial. Las expresiones regulares básicas también admiten cuantificadores `+` y `?`, Pero deben ir precedidas de una barra invertida. A diferencia de las expresiones regulares extendidas, `+` y `?` Por sí mismos son caracteres literales en expresiones regulares básicas.

Límites

Un *bound (límite)* es un cuantificador de átomos que, como su nombre indica, permite al usuario especificar límites cuantitativos precisos para un átomo. En las expresiones regulares extendidas, un límite puede aparecer de tres formas:

`{i}`

El átomo debe aparecer exactamente `i` veces (`i` un número entero). Por ejemplo, `[[:blank:]]{2}` coincide con exactamente dos caracteres en blanco.

`{i,}`

El átomo debe aparecer al menos `i` veces (`i` un número entero). Por ejemplo, `[[:blank:]]{2,}` coincide con cualquier secuencia de dos o más caracteres en blanco.

{i,j}

El átomo debe aparecer al menos `i` veces y como máximo `j` veces (`i` y `j` números enteros, `j` mayor que `i`). Por ejemplo, `xyz{2,4}` coincide con la cadena `xy` seguida de entre dos y cuatro caracteres `z`.

En cualquier caso, si una subcadena coincide con una expresión regular y una subcadena más larga que comienza en el mismo punto también coincide, se considerará la subcadena más larga.

Las expresiones regulares básicas también admiten límites, pero los delimitadores deben estar precedidos por `\`: `\{` y `\}`. Por sí mismos, `{` y `}` se interpretan como caracteres literales. Un `\{` seguido de un carácter que no sea un dígito es un carácter literal, no el comienzo de un límite.

Ramas y referencias posteriores

Las expresiones regulares básicas también difieren de las expresiones regulares extendidas en otro aspecto importante: una expresión regular extendida se puede dividir en *ramas*, cada una de las cuales es una expresión regular independiente. Las ramas están separadas por `|` y la expresión regular combinada coincidirá con cualquier cosa que corresponda a cualquiera de las ramas. Por ejemplo, `he|him` coincidirá si la subcadena `he` o `him` se encuentran en la cadena que se está examinando. Las expresiones regulares básicas interpretan `|` como un carácter literal. Sin embargo, la mayoría de los programas que soportan expresiones regulares básicas permitirán ramificaciones con `\|`.

Una expresión regular extendida encerrada entre `()` se puede usar en una *referencia posterior*. Por ejemplo, `([[:dígito:]])\1` coincidirá con cualquier expresión regular que se repita al menos una vez, porque el `\1` en la expresión es la referencia posterior a la pieza que coincide con la primera subexpresión entre paréntesis. Si existe más de una subexpresión entre paréntesis en la expresión regular, se puede hacer referencia a ellas con `\2`,`\3`, etc.

Para REs básicas, las subexpresiones deben estar delimitadas por `\(` y `\)`, con `(` y `)` por sí mismos caracteres ordinarios. El índice de referencia inversa se usa del mismo modo que en las expresiones regulares extendidas.

Búsqueda con expresiones regulares

El beneficio inmediato que ofrecen las expresiones regulares es mejorar las búsquedas en sistemas de archivos y en documentos de texto. La opción `-regex` del comando `find` permite probar cada ruta en una jerarquía de directorios contra una expresión regular. Por ejemplo,

```
$ find $HOME -regex '.*/\..*' -size +100M
```

busca archivos de más de 100 megabytes (100 unidades de 1048576 bytes), pero sólo en rutas dentro del directorio de inicio del usuario que contienen una coincidencia con `.*/\..*`, es decir, un `/.` rodeado por cualquier otro número de caracteres. En otras palabras, sólo se enumerarán los archivos ocultos o los archivos dentro de los directorios ocultos, independientemente de la posición de `/.` en la ruta correspondiente. Para expresiones regulares que no distinguen entre mayúsculas y minúsculas, se usará en su lugar la opción `-iregex`:

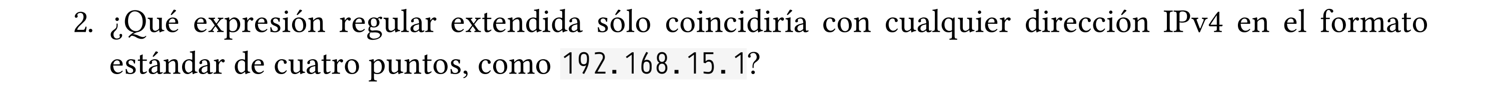
```
$ find /usr/share/fonts -regextype posix-extended -iregex
'.*(dejavu|liberation).*sans.*(italic|oblique).*'
/usr/share/fonts/dejavu/DejaVuSansCondensed-BoldOblique.ttf
/usr/share/fonts/dejavu/DejaVuSansCondensed-Oblique.ttf
/usr/share/fonts/dejavu/DejaVuSans-BoldOblique.ttf
/usr/share/fonts/dejavu/DejaVuSans-Oblique.ttf
/usr/share/fonts/dejavu/DejaVuSansMono-BoldOblique.ttf
/usr/share/fonts/dejavu/DejaVuSansMono-Oblique.ttf
/usr/share/fonts/liberation/LiberationSans-BoldItalic.ttf
/usr/share/fonts/liberation/LiberationSans-Italic.ttf
```

En este ejemplo, la expresión regular contiene ramas (escritas en modo *extendido*) para listar sólo archivos de fuentes específicos bajo la jerarquía de directorios `/usr/share/fonts`. Las expresiones regulares extendidas no son compatibles de forma predeterminada, pero `find` permite habilitarlas con `-regextype posix-extended` o `-regextype egrep`. El estándar RE predeterminado para `find` es *findutils-default*, que es prácticamente un clon básico de expresión regular.

A menudo es necesario pasar la salida de un programa al comando `less` cuando no cabe en la pantalla. El comando `less` divide su entrada en páginas, una pantalla completa a la vez, lo que permite al usuario navegar fácilmente por el texto hacia arriba y hacia abajo. Además, `less` también permite al usuario realizar búsquedas basadas en expresiones regulares. Esta característica es notablemente importante porque `less` es el paginador predeterminado que se usa para muchas tareas diarias, como inspeccionar entradas de diario o consultar páginas de manual. Al leer una página de manual, por ejemplo, al presionar la tecla / se abrirá un mensaje de búsqueda. Este es un escenario típico en el que las expresiones regulares son útiles, ya que las opciones de comando se enumeran justo después de un margen de página. Sin embargo, la misma opción puede aparecer muchas veces en el texto, lo que hace que las búsquedas literales sean inviables. Independientemente de eso, al escribir `^[[:blank:]]-o —  o de manera más simple:`^-o` — en el indicador de búsqueda y presionar Enter se saltará inmediatamente a la sección `-o` (si existe), permitiendo consultar la descripción de una opción de manera más rápida.

Ejercicios Guiados

1. ¿Qué expresión regular extendida coincidiría con cualquier dirección de correo electrónico, como `info@example.org`?



2. ¿Qué expresión regular extendida sólo coincidiría con cualquier dirección IPv4 en el formato estándar de cuatro puntos, como `192.168.15.1`?

3. ¿Cómo se puede usar el comando `grep` para listar el contenido del archivo `/etc/services`, descartando todos los comentarios (líneas que comienzan con `#`)?

4. El archivo `domains.txt` contiene una lista de nombres de dominio, uno por línea. ¿Cómo se usaría el comando `egrep` para listar solo los dominios `.org` o `.com`?

Ejercicios Exploratorios

1. Desde el directorio actual, ¿cómo usaría el comando `find` con una expresión regular extendida para buscar todos los archivos que no contienen un sufijo de archivo estándar (los nombres de archivo que no terminan en `.txt` o `.c`, por ejemplo)?

2. El comando `less` es el paginador predeterminado para mostrar archivos de texto largos en el entorno de shell. Al escribir `/`, se puede ingresar una expresión regular en el campo de búsqueda para saltar a la primera coincidencia correspondiente. Para permanecer en la posición actual del documento y sólo resaltar las coincidencias correspondientes, ¿qué combinación de teclas se debe ingresar en el campo de búsqueda?

3. Con `less`, ¿cómo sería posible filtrar la salida para que sólo se muestren las líneas que coinciden con una expresión regular?

Resumen

Esta lección cubre el soporte general de Linux para expresiones regulares, un estándar ampliamente utilizado cuyas capacidades de búsqueda de patrones son compatibles con la mayoría de los programas relacionados con texto. La lección pasa por los siguientes pasos:

- ¿Qué es una expresión regular?
- Los componentes principales de una expresión regular.
- Las diferencias entre expresiones regulares regulares y extendidas.
- ¿Cómo realizar búsquedas simples de texto y archivos usando expresiones regulares?

Respuestas a los ejercicios guiados

1. ¿Qué expresión regular extendida coincidiría con cualquier dirección de correo electrónico, como `info@example.org`?

```
egrep "\S+@\S+\.\S+"
```

2. ¿Qué expresión regular extendida sólo coincidiría con cualquier dirección IPv4 en el formato estándar de cuatro puntos, como `192.168.15.1`?

```
egrep "[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}"
```

3. ¿Cómo se puede usar el comando `grep` para listar el contenido del archivo `/etc/services`, descartando todos los comentarios (líneas que comienzan con `#`)?

```
grep -v ^# /etc/services
```

4. El archivo `domains.txt` contiene una lista de nombres de dominio, uno por línea. ¿Cómo se usaría el comando `egrep` para listar solo los dominios `.org` o `.com`?

```
egrep ".org$|.com$" domains.txt
```

Respuestas a ejercicios exploratorios

1. Desde el directorio actual, ¿cómo usaría el comando `find` con una expresión regular extendida para buscar todos los archivos que no contienen un sufijo de archivo estándar (los nombres de archivo que no terminan en `.txt` o `.c`, por ejemplo)?

```
find . -type f -regextype egrep -not -regex '.*\.[[:alnum:]]{1,}$'
```

2. El comando `less` es el paginador predeterminado para mostrar archivos de texto largos en el entorno de shell. Al escribir `/`, se puede ingresar una expresión regular en el campo de búsqueda para saltar a la primera coincidencia correspondiente. Para permanecer en la posición actual del documento y sólo resaltar las coincidencias correspondientes, ¿qué combinación de teclas se debe ingresar en el campo de búsqueda?

   Presionando `Ctrl` + `K` antes de ingresar la expresión de búsqueda.

3. Con `less`, ¿cómo sería posible filtrar la salida para que sólo se muestren las líneas que coinciden con una expresión regular?

   Presionando `&` e ingresando la expresión de búsqueda.



103.7 Lección 2

Certificación:	LPIC-1
Versión:	5.0
Tema:	103 Comandos GNU y Unix
Objetivo:	103.7 Buscar archivos de texto usando expresiones regulares
Lección:	2 de 2

Introducción

La transmisión de datos a través de una cadena de comandos canalizados permite la aplicación de filtros compuestos basados en expresiones regulares. Las expresiones regulares son una técnica importante que se utiliza no sólo en la administración de sistemas, sino también en la minería de datos y áreas relacionadas. Dos comandos son especialmente adecuados para manipular archivos y datos de texto usando expresiones regulares: `grep` y `sed`. `grep` es un buscador de patrones y `sed` es un editor de flujo. Son útiles por sí mismos, pero destacan todavía más cuando se usan en conjunción.

El buscador de patrones: grep

Uno de los usos más comunes de `grep` es facilitar la inspección de archivos largos, utilizando la expresión regular como filtro aplicado a cada línea. Se puede usar para mostrar sólo las líneas que comienzan con un término determinado. Por ejemplo, `grep` se puede usar para inspeccionar un archivo de configuración con módulos del kernel, mostrando sólo las líneas de opciones:

```
$ grep '^options' /etc/modprobe.d/alsa-base.conf
options snd-pcsp index=-2
options snd-usb-audio index=-2
options bt87x index=-2
options cx88_alsa index=-2
options snd-atiixp-modem index=-2
options snd-intel8x0m index=-2
options snd-via82xx-modem index=-2
```

El carácter de barra vertical `|` se puede emplear para redirigir la salida de un comando directamente a la entrada de `grep`. El siguiente ejemplo usa una expresión entre corchetes para seleccionar líneas de la salida de `fdisk -l`, comenzando con `Disk /dev/sda` o `Disk /dev/sdb`:

```
# fdisk -l | grep '^Disk /dev/sd[ab]'
Disk /dev/sda: 320.1 GB, 320072933376 bytes, 625142448 sectors
Disk /dev/sdb: 7998 MB, 7998537728 bytes, 15622144 sectors
```

La mera selección de líneas con coincidencias puede no ser apropiada para una tarea en particular, requiriendo ajustes en el comportamiento de `grep` a través de sus opciones. Por ejemplo, la opción `-c` o `--count` le dice a `grep` que muestre cuántas líneas tenían coincidencias:

```
# fdisk -l | grep '^Disk /dev/sd[ab]' -c
2
```

La opción se puede colocar antes o después de la expresión regular. Otras opciones importantes de grep son:

`-c` o `--count`
: En lugar de mostrar los resultados de la búsqueda, sólo muestra el recuento total de cuántas veces se produce una coincidencia en un archivo determinado.

`-i` o `--ignore-case`
: Hace que la búsqueda no distinga entre mayúsculas y minúsculas.

`-f FILE` o `--file=FILE`
: Indica un archivo que contenga la expresión regular a utilizar.

`-n` o `--line-number`

Muestra el número de la línea.

`-v` o `--invert-match`

Seleccioa todas las líneas, excepto las que contengan coincidencias.

`-H` o `--with-filename`

Imprime también el nombre del archivo que contiene la línea.

`-z` o `--null-data`

En lugar de que `grep` trate los flujos de datos de entrada y salida como líneas separadas (usando *newline* por defecto), toma la entrada o salida como una secuencia de líneas. Cuando se combina la salida del comando `find` usando su opción `-print0` con el comando `grep`, la opción `-z` o `--null-data` debe usarse para procesar el flujo de la misma manera.

Aunque se activa de forma predeterminada cuando se proporcionan varias rutas de archivo como entrada, la opción `-H` no está activada para archivos individuales. Eso puede ser crítico en situaciones especiales, como cuando `grep` es llamado directamente por `find`, por ejemplo:

```
$ find /usr/share/doc -type f -exec grep -i '3d modeling' "{}" \; | cut -c -100
artistic aspects of 3D modeling. Thus this might be the application you are
This major approach of 3D modeling has not been supported
oce is a C++ 3D modeling library. It can be used to develop CAD/CAM softwares, for
instance [FreeCad
```

En este ejemplo, `find` lista todos los archivos en `/usr/share/doc` y luego pasa cada uno a `grep`, que a su vez realiza una búsqueda que no distingue entre mayúsculas y minúsculas de `3d modeling` dentro del archivo. La tubería (*pipe*) está ahí sólo para limitar la longitud de salida a 100 columnas. Sin embargo, tenga en cuenta que no hay forma de saber de qué archivo provienen las líneas. Este problema se resuelve agregando `-H` a `grep`:

```
$ find /usr/share/doc -type f -exec grep -i -H '3d modeling' "{}" \; | cut -c -100
/usr/share/doc/openscad/README.md:artistic aspects of 3D modeling. Thus this might
be the applicatio
/usr/share/doc/opencsg/doc/publications.html:This major approach of 3D modeling has
not been support
```

Ahora es posible identificar los archivos donde se encontró cada coincidencia. Para que la lista sea aún más informativa, se pueden agregar líneas iniciales y finales a las líneas con coincidencias:

```
$ find /usr/share/doc -type f -exec grep -i -H -1 '3d modeling' "{}" \; | cut -c
-100
/usr/share/doc/openscad/README.md-application Blender), OpenSCAD focuses on the CAD
aspects rather t
/usr/share/doc/openscad/README.md:artistic aspects of 3D modeling. Thus this might
be the applicatio
/usr/share/doc/openscad/README.md-looking for when you are planning to create 3D
models of machine p
/usr/share/doc/opencsg/doc/publications.html-3D graphics library for Constructive
Solid Geometry (CS
/usr/share/doc/opencsg/doc/publications.html:This major approach of 3D modeling has
not been support
/usr/share/doc/opencsg/doc/publications.html-by real-time computer graphics until
recently.
```

La opción `-1` le indica a `grep` que incluya una línea antes y una línea después cuando encuentre una línea con una coincidencia. Estas líneas adicionales se denominan *líneas de contexto* y se identifican en la salida con un signo *menos* después del nombre del archivo. Se puede obtener el mismo resultado con `-C 1` o `--context=1` y se pueden indicar otras cantidades de líneas de contexto.

Hay dos programas complementarios a grep: `egrep` y `fgrep`. El programa `egrep` es equivalente al comando `grep -E`, que incorpora características adicionales además de las expresiones regulares básicas. Por ejemplo, con `egrep` es posible usar características extendidas de expresión regular, como ramificaciones:

```
$ find /usr/share/doc -type f -exec egrep -i -H -1 '3d (modeling|printing)' "{}" \;
| cut -c -100
/usr/share/doc/openscad/README.md-application Blender), OpenSCAD focuses on the CAD
aspects rather t
/usr/share/doc/openscad/README.md:artistic aspects of 3D modeling. Thus this might
be the applicatio
/usr/share/doc/openscad/README.md-looking for when you are planning to create 3D
models of machine p
/usr/share/doc/openscad/RELEASE_NOTES.md-* Support for using 3D-Mouse / Joystick /
Gamepad input dev
/usr/share/doc/openscad/RELEASE_NOTES.md:* 3D Printing support: Purchase from a
print service partne
/usr/share/doc/openscad/RELEASE_NOTES.md-* New export file formats: SVG, 3MF, AMF
/usr/share/doc/opencsg/doc/publications.html-3D graphics library for Constructive
Solid Geometry (CS
/usr/share/doc/opencsg/doc/publications.html:This major approach of 3D modeling has
not been support
/usr/share/doc/opencsg/doc/publications.html-by real-time computer graphics until
recently.
```

En este ejemplo, `3D modeling` o `3D printing` coincidirán con la expresión, no distingue entre mayúsculas y minúsculas. Para mostrar sólo las partes de un flujo de texto que coinciden con la expresión utilizada por `egrep`, use la opción `-o`.

El programa `fgrep` es equivalente a `grep -F`, es decir, no analiza expresiones regulares. Es útil en búsquedas simples donde el objetivo es hacer coincidir una expresión literal. Por lo tanto, los caracteres especiales como el signo de *dólar* y el *punto* se tomarán literalmente y no por su significado en una expresión regular.

El editor de transmisiones: sed

El propósito del programa `sed` es modificar datos basados en texto de una manera no interactiva. Significa que toda la edición se realiza mediante instrucciones predefinidas, no escribiendo arbitrariamente directamente en un texto que se muestra en la pantalla. En términos modernos, `sed` puede entenderse como un analizador de plantillas: dado un texto como entrada, coloca contenido personalizado en posiciones predefinidas o cuando encuentra una coincidencia para una expresión regular.

Sed, como su nombre indica, es muy adecuado para texto transmitido a través de *tuberías*. Su sintaxis básica es `sed -f SCRIPT` cuando las instrucciones de edición se almacenan en el archivo `SCRIPT` o `sed -e COMMANDS` para ejecutar `COMMANDS` directamente desde la línea de comandos. Si no hay `-f` ni `-e`, `sed` utiliza el primer parámetro que no es una opción como archivo de script. También es posible usar un

archivo como entrada simplemente dando su ruta como argumento a `sed`.

Las instrucciones `sed` se componen de un solo carácter, posiblemente precedidas por una dirección o seguidas de una o más opciones, y se aplican a cada línea a la vez. Las direcciones pueden ser un número de una sola línea, una expresión regular o un rango de líneas. Por ejemplo, la primera línea de una secuencia de texto se puede eliminar con `1d`, donde `1` especifica la línea donde se aplicará el comando de eliminación `d`. Para aclarar el uso de `sed`, tome la salida del comando `` factor `seq 12` ``, que devuelve los factores primos de los números del 1 al 12:

```
$ factor `seq 12`
1:
2: 2
3: 3
4: 2 2
5: 5
6: 2 3
7: 7
8: 2 2 2
9: 3 3
10: 2 5
11: 11
12: 2 2 3
```

La eliminación de la primera línea con `sed` se logra mediante `1d`:

```
$ factor `seq 12` | sed 1d
2: 2
3: 3
4: 2 2
5: 5
6: 2 3
7: 7
8: 2 2 2
9: 3 3
10: 2 5
11: 11
12: 2 2 3
```

Se puede especificar un rango de líneas con una coma separadora:

```
$ factor `seq 12` | sed 1,7d
8: 2 2 2
9: 3 3
10: 2 5
11: 11
12: 2 2 3
```

Se puede utilizar más de una instrucción en la misma ejecución, separadas por punto y coma. En este caso, sin embargo, es importante encerrarlos entre paréntesis para que el punto y coma no sea interpretado por el shell:

```
$ factor `seq 12` | sed "1,7d;11d"
8: 2 2 2
9: 3 3
10: 2 5
12: 2 2 3
```

En este ejemplo, se ejecutaron dos instrucciones de eliminación, primero en las líneas que van del 1 al 7 y luego en la línea 11. Una dirección también puede ser una expresión regular, por lo que sólo las líneas con una coincidencia se verán afectadas por la instrucción:

```
$ factor `seq 12` | sed "1d;/:.*2.*/d"
3: 3
5: 5
7: 7
9: 3 3
11: 11
```

La expresión regular `:.*2.*` coincide con cualquier aparición del número 2 en cualquier lugar después de dos puntos, lo que provoca la eliminación de las líneas correspondientes a los números con 2 como factor. Con `sed`, cualquier cosa colocada entre barras (/) se considera una expresión regular y, por defecto, se admiten todos los RE básicos. Por ejemplo, `sed -e "/^#/d" /etc/services` muestra el contenido del archivo `/etc/services` sin las líneas que comienzan con `#` (líneas de comentarios).

La instrucción de eliminación `d` es sólo una de las muchas instrucciones de edición proporcionadas por` sed`. En lugar de eliminar una línea, `sed` puede reemplazarla con un texto dado:

```
$ factor `seq 12` | sed "1d;/:.*2.*/c REMOVED"
REMOVED
3: 3
REMOVED
5: 5
REMOVED
7: 7
REMOVED
9: 3 3
REMOVED
11: 11
REMOVED
```

La instrucción `c REMOVED` simplemente reemplaza una línea con el texto `REMOVED`. En el caso del ejemplo, cada línea con una subcadena que coincida con la expresión regular `:.*2.*` se ve afectada por la instrucción `c REMOVED`. La instrucción `a TEXT` copia el texto indicado por `TEXT` en una nueva línea después de la línea con una coincidencia. La instrucción `r FILE` hace lo mismo, pero copia el contenido del archivo indicado por `FILE`. La instrucción `w` hace lo contrario de `r`, es decir, la línea se agregará al archivo indicado.

Con mucho, la instrucción `sed` más utilizada es `s/FIND/REPLACE/`, que se utiliza para reemplazar una coincidencia de la expresión regular `FIND` con el texto indicado por `REPLACE`. Por ejemplo, la instrucción `s/hda/sda/` reemplaza una subcadena que coincide con el literal RE `hda` con `sda`. Sólo se reemplazará la primera coincidencia que se encuentre en la línea, a menos que la opción `g` se coloque después de la instrucción, como en `s/hda/sda/g`.

Un estudio de caso más realista ayudará a ilustrar las características de `sed`. Suponga que una clínica médica desea enviar mensajes de texto a sus clientes, recordándoles sus citas programadas para el día siguiente. Un escenario de implementación típico se basa en un servicio de mensajería instantánea profesional, que proporciona una API para acceder al sistema responsable de entregar los mensajes. Estos mensajes generalmente se originan en el mismo sistema que ejecuta la aplicación que controla las citas del cliente, activados a una hora específica del día o algún otro evento. En esta situación hipotética, la aplicación podría generar un archivo llamado `citas.csv` que contenga datos tabulados con todas las citas para el día siguiente, luego usado por `sed` para procesar los mensajes de texto de un archivo de plantilla llamado `template.txt` . Los archivos CSV son una forma estándar de exportar datos de consultas de bases de datos, por lo que las citas de muestra se pueden proporcionar de la siguiente manera:

```
$ cat appointments.csv
"NAME","TIME","PHONE"
"Carol","11am","55557777"
"Dave","2pm","33334444"
```

La primera línea contiene las etiquetas de cada columna, que se utilizarán para hacer coincidir las etiquetas dentro del archivo de plantilla de muestra:

```
$ cat template.txt
Hey <NAME>, don't forget your appointment tomorrow at <TIME>.
```

Los signos de menos de `<` y de mayor que `>` se colocaron alrededor de las etiquetas sólo para ayudar a identificarlas como etiquetas. El siguiente script de Bash analiza todas las citas en cola usando `template.txt` como plantilla de mensaje:

```
#! /bin/bash

TEMPLATE=`cat template.txt`
TAGS=(`sed -ne '1s/^"//;1s/","/\n/g;1s/"$//p' appointments.csv`)
mapfile -t -s 1 ROWS < appointments.csv
for (( r = 0; r < ${#ROWS[*]}; r++ ))
do
  MSG=$TEMPLATE
  VALS=(`sed -e 's/^"//;s/","/\n/g;s/"$//' <<<${ROWS[$r]}`)
  for (( c = 0; c < ${#TAGS[*]}; c++ ))
  do
    MSG=`sed -e "s/<${TAGS[$c]}>/${VALS[$c]}/g" <<<"$MSG"`
  done
  echo curl --data message=\"$MSG\" --data phone=\"${VALS[2]}\"
https://mysmsprovider/api
done
```

Un script de producción real también controlaría la autenticación, la verificación de errores y el registro, pero se trata de un ejemplo con funcionalidad básica para empezar. Las primeras instrucciones ejecutadas por `sed` se aplican sólo a la primera línea — the address `1` in `1s/^"//;1s/","/\n/g;1s/"$//p` — para eliminar las comillas iniciales y finales — `1s/^"//` y `1s/"$//` — y reemplazar los separadores de campo con un carácter de nueva línea: `1s/","/\n/g`. Solo se necesita la primera línea para cargar los nombres de las columnas, por lo que las líneas que no coincidan se suprimirán con la opción `-n`, lo que requiere que la marca `p` se coloque después del último comando `sed` para imprimir la línea coincidente. Luego, las etiquetas se almacenan en la

variable `TAGS` como una matriz Bash. Otra variable de tipo matriz Bash es creada por el comando `mapfile` para almacenar las líneas que contienen las citas en la variable de matriz `ROWS`.

Se emplea un bucle `for` para procesar cada línea de cita que se encuentra en `ROWS`. Luego, las comillas y los separadores en la cita - la cita está en la variable `${ROWS[$r]}` usada como una *here string* - se reemplazan por `sed`, de manera similar a los comandos usados para cargar las etiquetas. Los valores separados para la cita se almacenan en la variable de matriz `VALS`, donde los subíndices de matriz 0, 1 y 2 corresponden a los valores de `NAME`, `TIME` y `PHONE`.

Finalmente, un bucle anidado `for` recorre la matriz `TAGS` y reemplaza cada etiqueta que se encuentra en la plantilla con su valor correspondiente en `VALS`. La variable `MSG` contiene una copia de la plantilla renderizada, actualizada por el comando de sustitución `s/<${TAGS[$c]}>/${VALS[$c]}/g` en cada iteración de bucle a través de `TAGS`.

Esto da como resultado un mensaje como: `"Hey Carol, don’t forget your appointment tomorrow at 11am."` El mensaje se puede enviar como un parámetro a través de una solicitud HTTP con `curl`, como un mensaje de correo o cualquier otro método similar.

Combinando grep y sed

Los comandos `grep` y `sed` pueden usarse juntos cuando se requieren procedimientos de minería de texto más complejos. Como administrador del sistema, es posible que desee inspeccionar todos los intentos de inicio de sesión en un servidor, por ejemplo. El archivo `/var/log/wtmp` registra todos los inicios y cierres de sesión, mientras que el archivo `/var/log/btmp` registra los intentos fallidos de inicio de sesión. Están escritos en formato binario, que se pueden leer con los comandos `last` y `lastb`, respectivamente.

La salida de `lastb` muestra no solo el nombre de usuario utilizado en el intento de inicio de sesión incorrecto, sino también su dirección IP:

```
# lastb -d -a -n 10 --time-format notime
user     ssh:notty       (00:00)     81.161.63.251
nrostagn ssh:notty       (00:00)     vmd60532.contaboserver.net
pi       ssh:notty       (00:00)     132.red-88-20-39.staticip.rima-tde.net
pi       ssh:notty       (00:00)     132.red-88-20-39.staticip.rima-tde.net
pi       ssh:notty       (00:00)     46.6.11.56
pi       ssh:notty       (00:00)     46.6.11.56
nps      ssh:notty       (00:00)     vmd60532.contaboserver.net
narmadan ssh:notty       (00:00)     vmd60532.contaboserver.net
nominati ssh:notty       (00:00)     vmd60532.contaboserver.net
nominati ssh:notty       (00:00)     vmd60532.contaboserver.net
```

La opción `-d` traduce el número de IP al nombre de host correspondiente. El nombre de host puede proporcionar pistas sobre el ISP o el servicio de alojamiento utilizado para realizar estos intentos de inicio de sesión incorrectos. La opción `-a` coloca el nombre de host en la última columna, lo que facilita el filtrado aún por aplicar. La opción `--time-format notime` suprime la hora en que se produjo el intento de inicio de sesión. El comando `lastb` puede tardar algún tiempo en completarse si hubo demasiados intentos de inicio de sesión incorrectos, por lo que la salida se limitó a diez entradas con la opción `-n 10`.

No todas las direcciones IP remotas tienen un nombre de host asociado, por lo que el DNS inverso no se aplica a ellas y se pueden descartar. Aunque se podría escribir una expresión regular para que coincida con el formato esperado para un nombre de host al final de la línea, probablemente sea más sencillo escribir una expresión regular para que coincida con una letra del alfabeto o con un solo dígito al final de la línea. El siguiente ejemplo muestra cómo el comando `grep` toma el listado en su entrada estándar y elimina las líneas sin nombre de host:

```
# lastb -d -a --time-format notime | grep -v '[0-9]$' | head -n 10
nvidia   ssh:notty       (00:00)     vmd60532.contaboserver.net
n_tonson ssh:notty       (00:00)     vmd60532.contaboserver.net
nrostagn ssh:notty       (00:00)     vmd60532.contaboserver.net
pi       ssh:notty       (00:00)     132.red-88-20-39.staticip.rima-tde.net
pi       ssh:notty       (00:00)     132.red-88-20-39.staticip.rima-tde.net
nps      ssh:notty       (00:00)     vmd60532.contaboserver.net
narmadan ssh:notty       (00:00)     vmd60532.contaboserver.net
nominati ssh:notty       (00:00)     vmd60532.contaboserver.net
nominati ssh:notty       (00:00)     vmd60532.contaboserver.net
nominati ssh:notty       (00:00)     vmd60532.contaboserver.net
```

El comando `grep` con la opción `-v`, muestra solo las líneas que no coinciden con la expresión regular dada. Una expresión regular que coincida con cualquier línea que termine con un número (es decir, `[0-9]$`) capturará solo las entradas sin un nombre de host. Por lo tanto, `grep -v '[0-9]$'` mostrará solo las líneas que terminan con un nombre de host.

La salida se puede filtrar aún más, manteniendo solo el nombre de dominio y eliminando las otras partes de cada línea. El comando `sed` puede hacerlo con un comando de sustitución para reemplazar toda la línea con una referencia al nombre de dominio en ella:

```
# lastb -d -a --time-format notime | grep -v '[0-9]$' | sed -e 's/.* \(.*\)$/\1/' |
head -n 10
vmd60532.contaboserver.net
vmd60532.contaboserver.net
vmd60532.contaboserver.net
132.red-88-20-39.staticip.rima-tde.net
132.red-88-20-39.staticip.rima-tde.net
vmd60532.contaboserver.net
vmd60532.contaboserver.net
vmd60532.contaboserver.net
vmd60532.contaboserver.net
vmd60532.contaboserver.net
```

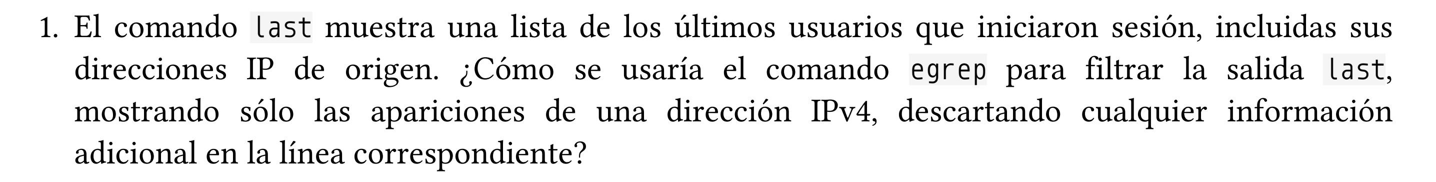
El paréntesis de escape en `.* \(.*\)$` indica a `sed` que recuerde esa parte de la línea, es decir, la parte entre el último carácter de espacio y el final de la línea. En el ejemplo, se hace referencia a esta parte con `\1` y se usa para reemplazar la línea completa.

Está claro que la mayoría de los hosts remotos intentan iniciar sesión más de una vez, por lo que el mismo nombre de dominio se repite. Para suprimir las entradas repetidas, primero se deben ordenar (con el comando `sort`) y luego pasar al comando `uniq`:

```
# lastb -d -a --time-format notime | grep -v '[0-9]$' | sed -e 's/.* \(.*\)$/\1/' |
sort | uniq | head -n 10
116-25-254-113-on-nets.com
132.red-88-20-39.staticip.rima-tde.net
145-40-33-205.power-speed.at
tor.laquadrature.net
tor.momx.site
ua-83-226-233-154.bbcust.telenor.se
vmd38161.contaboserver.net
vmd60532.contaboserver.net
vmi488063.contaboserver.net
vmi515749.contaboserver.net
```

Esto muestra cómo se pueden combinar diferentes comandos para producir el resultado deseado. La lista de nombres de host se puede utilizar para escribir reglas de bloqueo de firewall o para tomar otras medidas para hacer cumplir la seguridad del servidor.

Ejercicios Guiados



1. El comando `last` muestra una lista de los últimos usuarios que iniciaron sesión, incluidas sus direcciones IP de origen. ¿Cómo se usaría el comando `egrep` para filtrar la salida `last`, mostrando sólo las apariciones de una dirección IPv4, descartando cualquier información adicional en la línea correspondiente?

2. ¿Qué opción se le debe pasar a `grep` para filtrar correctamente la salida generada por el comando `find` ejecutado con la opción `-print0`?

3. El comando `uptime -s` muestra la última fecha en la que se encendió el sistema, como en `2019-08-05 20:13:22`. ¿Cuál será el resultado del comando `uptime -s | sed -e 's/(.*) (.*)/\1/'`?

4. ¿Qué opción se le debe pasar a `grep` para que cuente las líneas coincidentes en lugar de mostrarlas?

Ejercicios Exploratorios

1. La estructura básica de un archivo HTML comienza con los elementos `html`, `head` y `body`, por ejemplo:

```
<html>
<head>
  <title>News Site</title>
</head>
<body>
  <h1>Headline</h1>
  <p>Information of interest.</p>
</body>
</html>
```

   Describe cómo se pueden usar las direcciones en `sed` para mostrar sólo el elemento `body` y su contenido.

2. ¿Qué expresión `sed` eliminará todas las etiquetas de un documento HTML, manteniendo sólo el texto renderizado?

3. Los archivos con extensión `.ovpn` son muy populares para configurar clientes VPN ya que contienen no sólo la configuración, sino también el contenido de las claves y certificados para el cliente. Estas claves y certificados se encuentran originalmente en archivos separados, por lo que deben copiarse en el archivo `.ovpn`. Dado el siguiente extracto de una plantilla `.ovpn`:

```
client
dev tun
remote 192.168.1.155 1194
<ca>
ca.crt
</ca>
<cert>
client.crt
</cert>
<key>
client.key
</key>
<tls-auth>
ta.key
</tls-auth>
```

Suponiendo que los archivos `ca.crt`, `client.crt`, `client.key` y `ta.key` están en el directorio actual, ¿cómo modificaría `sed` la configuración de la plantilla para reemplazar cada nombre de archivo por su contenido?

Resumen

Esta lección cubre los dos comandos de Linux más importantes relacionados con las expresiones regulares: `grep` y `sed`. Los scripts y los comandos compuestos se basan en `grep` y `sed` para realizar una amplia gama de tareas de filtrado y análisis de texto. La lección pasa por los siguientes pasos:

- ¿Cómo usar `grep` y sus variaciones como `egrep` y `fgrep`?
- ¿Cómo usar `sed` y sus instrucciones internas para manipular texto?
- Ejemplos de aplicaciones de expresión regular que utilizan `grep` y `sed`.

Respuestas a los ejercicios guiados

1. El comando `last` muestra una lista de los últimos usuarios que iniciaron sesión, incluidas sus direcciones IP de origen. ¿Cómo se usaría el comando `egrep` para filtrar la salida `last`, mostrando sólo las apariciones de una dirección IPv4, descartando cualquier información adicional en la línea correspondiente?

   `last -i | egrep -o '[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}'`

2. ¿Qué opción se le debe pasar a `grep` para filtrar correctamente la salida generada por el comando `find` ejecutado con la opción `-print0`?

   La opción `-z` o `--null-data`, como en `find . -print0 | grep -z expression`.

3. El comando `uptime -s` muestra la última fecha en la que se encendió el sistema, como en `2019-08-05 20:13:22`. ¿Cuál será el resultado del comando `uptime -s | sed -e 's/(.*) (.*)/\1/'`?

   Ocurrirá un error. De forma predeterminada, los paréntesis deben *escaparse* para usar referencias anteriores en `sed`.

4. ¿Qué opción se le debe pasar a `grep` para que cuente las líneas coincidentes en lugar de mostrarlas?

   Opción `-c`.

Respuestas a ejercicios exploratorios

1. La estructura básica de un archivo HTML comienza con los elementos `html`, `head` y `body`, por ejemplo:

   ```
   <html>
   <head>
     <title>News Site</title>
   </head>
   <body>
     <h1>Headline</h1>
     <p>Information of interest.</p>
   </body>
   </html>
   ```

   Describe cómo se pueden usar las direcciones en `sed` para mostrar sólo el elemento `body` y su contenido.

   Para mostrar sólo `body`, las direcciones deben ser `/<body>/,/<\/body>/`, como en `sed -n -e '/<body>/,/<\/body>/p'`. La opción `-n` se le pasa a `sed` por lo que no imprime líneas por defecto, de ahí el comando `p` al final de la expresión `sed` para imprimir líneas coincidentes.

2. ¿Qué expresión `sed` eliminará todas las etiquetas de un documento HTML, manteniendo sólo el texto renderizado?

   La expresión `sed` `s/<[^>]*>//g` reemplazará cualquier contenido encerrado en `<>` por una cadena vacía.

3. Los archivos con extensión `.ovpn` son muy populares para configurar clientes VPN ya que contienen no sólo la configuración, sino también el contenido de las claves y certificados para el cliente. Estas claves y certificados se encuentran originalmente en archivos separados, por lo que deben copiarse en el archivo `.ovpn`. Dado el siguiente extracto de una plantilla `.ovpn`:

```
client
dev tun
remote 192.168.1.155 1194
<ca>
ca.crt
</ca>
<cert>
client.crt
</cert>
<key>
client.key
</key>
<tls-auth>
ta.key
</tls-auth>
```

Suponiendo que los archivos `ca.crt`, `client.crt`, `client.key` y `ta.key` están en el directorio actual, ¿cómo modificaría `sed` la configuración de la plantilla para reemplazar cada nombre de archivo por su contenido?

El comando

```
sed -r -e 's/(^[^.]*)\.(crt|key)$/cat \1.\2/e' < client.template > client.ovpn
```

reemplaza cualquier línea que termine en `.crt` o `.key` por el contenido de un archivo cuyo nombre es igual a la línea. La opción `-r` indica a `sed` que use expresiones regulares extendidas, mientras que `e` al final de la expresión indica a `sed` que reemplace las coincidencias con la salida del comando `cat \1.\2`. Las referencias inversas `\1` y `\2` corresponden al nombre de archivo y la extensión encontrados en la coincidencia.



103.8 Edición básica de archivos

Referencia al objetivo del LPI

LPIC-1 v5, Exam 101, Objective 103.8

Importancia

3

Áreas de conocimiento clave

- Navegar por un documento usando vi.
- Entender y usar los modos de vi.
- Insertar, editar, borrar, copiar y encontrar texto usando vi.
- Conocimientos de Emacs, nano y vim.
- Configurar el editor estándar.

Lista parcial de archivos, términos y utilidades

- `vi`
- `/`, `?`
- `h`, `j`, `k`, `l`
- `i`, `o`, `a`
- `d`, `p`, `y`, `dd`, `yy`
- `ZZ`, `:w!`, `:q!`
- `EDITOR`



103.8 Lección 1

Certificación:	LPIC-1
Versión:	5.0
Tema:	103 Comandos GNU y Unix
Objetivo:	103.8 Edición básica de archivos
Lección:	1 de 1

Introducción

En la mayoría de las distribuciones de Linux, `vi`, abreviatura de “visual”, está preinstalado y es el editor estándar en el entorno de shell. Vi es un editor de texto interactivo, muestra el contenido del archivo en la pantalla mientras se está editando. Como tal, permite al usuario moverse y realizar modificaciones en cualquier parte del documento. Sin embargo, a diferencia de los editores visuales del escritorio gráfico, el editor `vi` es una aplicación de shell con atajos de teclado para cada tarea de edición.

Una alternativa a `vi`, llamada `vim` (*vi mejorado*), a veces se usa como un reemplazo moderno de `vi`. Entre otras mejoras, `vim` ofrece soporte para resaltado de sintaxis, deshacer/rehacer multinivel y edición de varios documentos. Aunque tiene más recursos, `vim` es totalmente compatible con `vi`, lo que hace que ambos sean indistinguibles para la mayoría de las tareas.

La forma estándar de iniciar `vi` es darle una ruta a un archivo como parámetro. Para saltar directamente a una línea específica, su número debe ser informado con un signo más, como en `vi +9 /etc/fstab` para abrir `/etc/fstab/` y colocar el cursor en la novena línea. Sin un número, el signo más por sí solo coloca el cursor en la última línea.

La interfaz de `vi` es muy simple: todo el espacio disponible en la ventana de la terminal está ocupado

para presentar un archivo, normalmente informado como un argumento de comando, al usuario. Las únicas pistas visuales son una línea de pie de página que muestra la posición actual del cursor y una tilde `~` que indica dónde termina el archivo. Hay diferentes modos de ejecución para `vi` donde cambia el comportamiento del programa. Los más comunes son: *modo inserción* y *modo normal*.

Modo de Inserción

El modo de inserción es sencillo: el texto aparece en la pantalla a medida que se escribe en el teclado. Es el tipo de interacción que la mayoría de los usuarios esperan de un editor de texto, pero no es la forma en que `vi` presenta un documento por primera vez. Para ingresar al modo de inserción, el usuario debe ejecutar un comando de inserción en el modo normal. La tecla *Esc* finaliza el modo de inserción y vuelve al modo normal, el modo `vi` predeterminado.

NOTE

Si está interesado en saber más sobre los otros modos de ejecución, abra `vi` y escriba:

```
:help vim-modes-intro
```

Modo Normal

El modo normal, también conocido como modo de comando, es cómo se inicia `vi` de forma predeterminada. En este modo, las teclas del teclado están asociadas con comandos para tareas de manipulación de texto y navegación. La mayoría de los comandos de este modo son teclas únicas. Algunas de las teclas y sus funciones en modo normal son:

`0`, `$`
Ve al principio y al final de la línea.

`1G`, `G`
Vaya al principio y al final del documento.

`(`, `)`
Vaya al principio y al final de la oración.

`{`, `}`
Vaya al principio y al final del párrafo.

`w`, `W`
Saltar palabra y saltar palabra, incluida la puntuación.

`h`, `j`, `k`, `l`
: Izquierda, abajo, arriba, derecha.

`e` o `E`
: Ir al final de la palabra actual.

`/`, `?`
: Busca hacia adelante y hacia atrás.

`i`, `I`
: Ingrese al modo de inserción antes de la posición actual del cursor y al comienzo de la línea actual.

`a`, `A`
: Ingrese al modo de inserción después de la posición actual del cursor y al final de la línea actual.

`o`, `O`
: Agregue una nueva línea e ingrese al modo de inserción en la línea siguiente o en la línea anterior.

`s`, `S`
: Borre el carácter debajo del cursor o toda la línea e ingrese al modo de inserción.

`c`
: Cambie el (los) carácter (es) debajo del cursor.

`r`
: Reemplaza el carácter debajo del cursor.

`x`
: Elimina los caracteres seleccionados o el carácter debajo del cursor.

`v`, `V`
: Inicie una nueva selección con el carácter actual o la línea completa.

`y`, `yy`
: Copia (tira) los caracteres o la línea completa.

`p, P`
: Pega el contenido copiado, antes o después de la posición actual.

`u`
: Deshace la última acción.

`Ctrl-R`
: Rehace la última acción.

`ZZ`
: Cerrar y guardar.

`ZQ`
: Cerrar y no guardar.

Si está precedido por un número, el comando se ejecutará el mismo número de veces. Por ejemplo, presione `3yy` para copiar la línea actual más las dos siguientes, presione `d5w` para eliminar la palabra actual y las 4 palabras siguientes, y así sucesivamente.

La mayoría de las tareas de edición son combinaciones de varios comandos. Por ejemplo, la secuencia de teclas `vey` se utiliza para copiar una selección desde la posición actual hasta el final de la palabra actual. La repetición de comandos también se puede usar en combinaciones, por lo que `v3ey` copiaría una selección comenzando en la posición actual hasta el final de la tercera palabra desde allí.

`vi` puede organizar el texto copiado en registros, lo que permite mantener distintos contenidos al mismo tiempo. Un registro se especifica con un carácter precedido por `"` y una vez creado se mantiene hasta el final de la sesión actual. La secuencia de teclas `"ly` crea un registro que contiene la selección actual, que será accesible a través de la tecla `l` . Luego, el registro `l` se puede pegar con `"lp`.

También hay una forma de establecer marcas personalizadas en posiciones arbitrarias a lo largo del texto, lo que facilita el salto rápido entre ellas. Las marcas se crean presionando la tecla `m` y luego una tecla para abordar la posición actual. Una vez hecho esto, el cursor volverá a la posición marcada cuando se presione `'` seguido de la tecla elegida.

Cualquier secuencia de teclas puede registrarse como una macro para su ejecución futura. Se puede grabar una macro, por ejemplo, para rodear un texto seleccionado entre comillas dobles. Primero, se selecciona una cadena de texto y se presiona la tecla `q`, seguida de una tecla de registro para asociar la macro, como `d`. La línea `recording @d` aparecerá en la línea del pie de página, indicando que la grabación está activada. Se supone que ya se ha seleccionado algún texto, por lo que el primer comando es `x` para eliminar (y copiar automáticamente) el texto seleccionado. Se presiona la tecla `i`

para insertar dos comillas dobles en la posición actual, luego `Esc` regresa al modo normal. El último comando es `P`, para volver a insertar la selección eliminada justo antes de la última comilla doble. Si presiona `q` nuevamente, finalizará la grabación. Ahora, una macro que consta de la secuencia de teclas `x`, `i`, `""`, `Esc` y `P` se ejecutará cada vez que se presionen las teclas `@d` en modo normal, donde `d` es la clave de registro asociada con la macro.

Sin embargo, la macro estará disponible solo durante la sesión actual. Para que las macros sean persistentes, deben almacenarse en el archivo de configuración. Como la mayoría de las distribuciones modernas usan *vim* como editor compatible con *vi*, el archivo de configuración del usuario es `~/.vimrc`. Dentro de `~/.vimrc`, la línea `let @d = 'xi"" P'` establecerá el registro `d` en la secuencia de teclas entre comillas simples. El mismo registro asignado previamente a una macro se puede utilizar para pegar su secuencia de teclas.

Comandos Colon

El modo normal también admite otro conjunto de comandos `vi`: los comandos *colon*. Los comandos de dos puntos, como su nombre indica, se ejecutan después de presionar la tecla de dos puntos `:` en modo normal. Los comandos de dos puntos permiten al usuario realizar búsquedas, guardar, salir, ejecutar comandos de shell, cambiar la configuración de `vi`, etc. Para volver al modo normal, se debe ejecutar el comando `:visual` o la tecla Enter presionado sin ningún comando. Aquí se indican algunos de los comandos de dos puntos más comunes (la inicial no es parte del comando):

`:s/REGEX/TEXT/g`

Reemplaza todas las apariciones de la expresión regular `REGEX` por `TEXT` en la línea actual. Acepta la misma sintaxis del comando `sed`, incluidas las direcciones.

`:!`

Ejecutar el comando de shell especificado a continuación.

`:quit` o **`:q`**

Salir del programa.

`:quit!` o **`:q!`**

Salir del programa sin guardar.

`:wq`

Guardar y Salir.

`:exit` o **`:x`** o **`:e`**

Guardar y salir, si es necesario.

`:visual`

Volver al modo de navegación.

El programa estándar `vi` es capaz de realizar la mayoría de las tareas de edición de texto, pero se puede usar cualquier otro editor no gráfico para editar archivos de texto en el entorno de shell.

TIP

Los usuarios novatos pueden tener dificultades para memorizar todas las teclas de comando de `vi` a la vez. Las distribuciones que adoptan `vim` también tienen el comando `vimtutor`, que usa `vim` en sí para abrir una guía paso a paso de las principales actividades. El archivo es una copia editable que se puede utilizar para practicar los comandos y progresivamente acostumbrarse a ellos.

Editores alternativos

Los usuarios que no estén familiarizados con `vi` pueden tener dificultades para adaptarse a él, ya que su funcionamiento no es intuitivo. Una alternativa más simple es GNU `nano`, un pequeño editor de texto que ofrece todas las funciones básicas de edición de texto como deshacer/rehacer, colorear sintaxis, búsqueda y reemplazo interactivos, sangría automática, números de línea, finalización de palabras, bloqueo de archivos, respaldo de archivos y apoyo a la internacionalización. A diferencia de `vi`, todas las pulsaciones de teclas se insertan en el documento que se está editando. Los comandos en `nano` se dan usando la tecla `Ctrl` o la tecla Meta (dependiendo del sistema, Meta es `Alt` o `▯`).

`Ctrl-6` o **`Meta-A`**

Iniciar una nueva selección. También es posible crear una selección presionando Shift y moviendo el cursor.

`Meta-6`

Copia la selección actual.

`Ctrl-K`

Cortar la selección actual.

`Ctrl-U`

Pegar el contenido copiado.

`Meta-U`

Deshacer.

`Meta-E`
Rehacer.

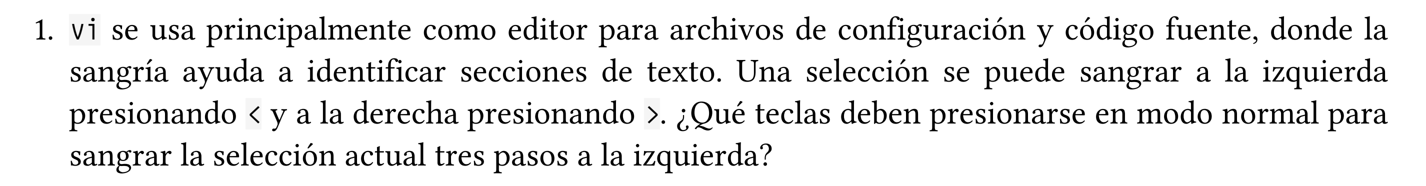
`Ctrl-\`
Reemplazar el texto en la selección.

`Ctrl-T`
Iniciar una sesión de revisión ortográfica para el documento o la selección actual.

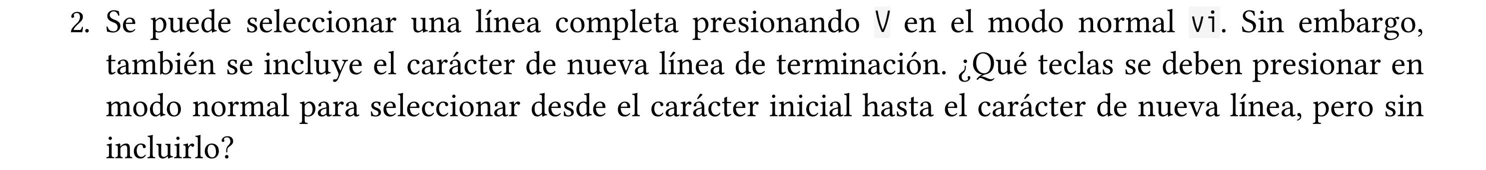
Emacs es otro editor de texto muy popular para el entorno de shell. Mientras que el texto se inserta simplemente escribiéndolo, como en `nano`, la navegación a través del documento es asistida por comandos de teclado, como en `vi`. Emacs incluye muchas características que lo convierten en algo más que un editor de texto. También es un IDE (*entorno de desarrollo integrado*) capaz de compilar, ejecutar y probar programas. Emacs se puede configurar como cliente de correo electrónico, noticias o RSS, lo que lo convierte en una auténtica suite productiva.

El propio shell ejecutará un editor de texto predeterminado, generalmente `vi`, cada vez que sea necesario. Este es el caso, por ejemplo, cuando se ejecuta `crontab -e` para editar *cronjobs*. Bash usa las variables de sesión `VISUAL` o `EDITOR` para encontrar el editor de texto predeterminado para el entorno de shell. Por ejemplo, el comando `export EDITOR=nano` define `nano` como el editor de texto predeterminado en la sesión de shell actual. Para que este cambio sea persistente en todas las sesiones, el comando debe incluirse en `~/.bash_profile`.

Ejercicios Guiados



1. `vi` se usa principalmente como editor para archivos de configuración y código fuente, donde la sangría ayuda a identificar secciones de texto. Una selección se puede sangrar a la izquierda presionando `<` y a la derecha presionando `>`. ¿Qué teclas deben presionarse en modo normal para sangrar la selección actual tres pasos a la izquierda?



2. Se puede seleccionar una línea completa presionando `V` en el modo normal `vi`. Sin embargo, también se incluye el carácter de nueva línea de terminación. ¿Qué teclas se deben presionar en modo normal para seleccionar desde el carácter inicial hasta el carácter de nueva línea, pero sin incluirlo?

3. ¿Cómo debería ejecutarse `vi` en la línea de comando para abrir `~/.bash_profile` y saltar directamente a la última línea?

4. ¿Qué teclas se deben presionar en el modo normal `vi` para eliminar caracteres desde la posición actual del cursor hasta el siguiente carácter de punto?

Ejercicios Exploratorios

1. `vim` permite seleccionar bloques de texto con ancho arbitrario, no sólo secciones con líneas completas. Al presionar Ctrl + V en modo normal, se realiza una selección moviendo el cursor hacia arriba, abajo, izquierda y derecha. Con este método, ¿cómo se eliminaría un bloque que comienza en el primer carácter de la línea actual, que contiene las siguientes ocho columnas y cinco líneas de texto?

2. Una sesión `vi` fue interrumpida por una falla de energía inesperada. Cuando se vuelve a abrir el archivo, `vi` pregunta al usuario si desea recuperar el archivo de intercambio (una copia automática realizada por `vi`). ¿Qué debe hacer el usuario para descartar el archivo de intercambio?

3. En una sesión de `vim`, una línea fue previamente copiada al registro `l`. ¿Qué combinación de teclas registraría una macro en el registro `a` para pegar la línea en el registro `l` inmediatamente antes de la línea actual?

Resumen

Esta lección cubre el editor de texto estándar para el entorno de shell de Linux: el editor `vi`. Aunque intimida al usuario desconocido, `vi` tiene características que lo convierten en una buena opción para la edición de texto técnica y no técnica. La lección pasa por los siguientes pasos:

- Uso básico de `vi` y funciones útiles.
- ¿Qué es `vim`? - el `vi` mejorado - y otros editores alternativos.
- ¿Cómo definir el editor de texto predeterminado para el entorno de shell?

Los comandos y procedimientos abordados fueron:

- Editor `vi` y su versión mejorada `vim`.
- Edición de texto básica en `vi`.
- Editores alternativos `emacs` y `nano`.

Respuestas a los ejercicios guiados

1. `vi` se usa principalmente como editor para archivos de configuración y código fuente, donde la sangría ayuda a identificar secciones de texto. Una selección se puede sangrar a la izquierda presionando `<` y a la derecha presionando `>`. ¿Qué teclas deben presionarse en modo normal para sangrar la selección actual tres pasos a la izquierda?

   Las teclas `3<`, es decir, tres pasos a la izquierda.

2. Se puede seleccionar una línea completa presionando `V` en el modo normal `vi`. Sin embargo, también se incluye el carácter de nueva línea de terminación. ¿Qué teclas se deben presionar en modo normal para seleccionar desde el carácter inicial hasta el carácter de nueva línea, pero sin incluirlo?

   Las teclas `0v$h`, que significan `0` ("saltar al inicio de una línea"), `v` ("iniciar la selección de caracteres"), `$` ("ir al final de la línea") y `h` ("retroceder una posición").

3. ¿Cómo debería ejecutarse `vi` en la línea de comando para abrir `~/.bash_profile` y saltar directamente a la última línea?

   El comando `vi + ~/.bash_profile` abrirá el archivo y colocará el cursor en su última línea.

4. ¿Qué teclas se deben presionar en el modo normal `vi` para eliminar caracteres desde la posición actual del cursor hasta el siguiente carácter de punto?

   Las teclas `dt`, que significan `d` ("iniciar eliminación"), `t` ("saltar al siguiente carácter") y `.` (carácter de punto).

Respuestas a ejercicios exploratorios

1. `vim` permite seleccionar bloques de texto con ancho arbitrario, no sólo secciones con líneas completas. Al presionar Ctrl + V en modo normal, se realiza una selección moviendo el cursor hacia arriba, abajo, izquierda y derecha. Con este método, ¿cómo se eliminaría un bloque que comienza en el primer carácter de la línea actual, que contiene las siguientes ocho columnas y cinco líneas de texto?

   La combinación `0`, `Ctrl-V` y `8l5jd` seleccionará y eliminará el bloque correspondiente.

2. Una sesión `vi` fue interrumpida por una falla de energía inesperada. Cuando se vuelve a abrir el archivo, `vi` pregunta al usuario si desea recuperar el archivo de intercambio (una copia automática realizada por `vi`). ¿Qué debe hacer el usuario para descartar el archivo de intercambio?

   Presione `d` cuando se lo solicite `vi`.

3. En una sesión de `vim`, una línea fue previamente copiada al registro `l`. ¿Qué combinación de teclas registraría una macro en el registro `a` para pegar la línea en el registro `l` inmediatamente antes de la línea actual?

   La combinación `qa"lPq`, que significa `q` (“iniciar grabación de macro”), `a` (“asignar registro `a` a macro”), `"l` (“seleccionar texto en registro `l`”), `P` (“pegar antes de la línea actual”) y `q` (“finalizar grabación macro”).



Tema 104: Dispositivos, sistemas de archivos Linux y el estándar de jerarquía de archivos



104.1 Creación de particiones y sistemas de archivos

Referencia al objetivo del LPI

LPIC-1 v5, Exam 101, Objective 104.1

Importancia

2

Áreas de conocimiento clave

- Administrar tablas de particiones MBR y GPT
- Usar diversos comandos mkfs para crear distintos sistemas de archivos tales como:
  - ext2/ext3/ext4
  - XFS
  - VFAT
  - exFAT
- Conocimientos básicos del sistema de archivos Btrfs, incluyendo los sistemas de archivos multidispositivo, la compresión y los subvolúmenes.

Lista parcial de archivos, términos y utilidades

- `fdisk`
- `gdisk`
- `parted`
- `mkfs`
- `mkswap`



104.1 Lección 1

Certificación:	LPIC-1
Versión:	5.0
Tema:	104 Dispositivos, sistemas de archivos Linux, estándar de jerarquía del sistema de archivos
Objetivo:	104.1 Crear particiones y sistemas de archivos
Lección:	1 de 1

Introducción

En cualquier sistema operativo, es necesario particionar un disco antes de poder usarlo. Una partición es un subconjunto lógico del disco físico y la información sobre las particiones se almacena en una tabla de particiones. Esta tabla incluye información sobre el primer y último sector de la partición y su tipo, y más detalles sobre cada partición.

Por lo general, un sistema operativo considera cada partición como un "disco" separado, incluso si todas residen en el mismo medio físico. En los sistemas Windows se les asignan letras como `C:` (históricamente el disco principal), `D:` y así sucesivamente. En Linux, cada partición se asigna a un directorio en `/dev`, como `/dev/sda1` o `/dev/sda2`.

En esta lección, aprenderá cómo crear, eliminar, restaurar y cambiar el tamaño de las particiones usando las tres utilidades más comunes (`fdisk`, `gdisk` y `parted`), cómo crear un sistema de archivos en ellas y cómo crear y configurar una *partición de intercambio* o *archivo de intercambio* para usar como memoria virtual.

NOTE Por razones históricas, a lo largo de esta lección nos referiremos a los medios de almacenamiento como “discos”, aunque los sistemas de almacenamiento modernos, como los SSD y el almacenamiento Flash, no contienen ningún “disco” en absoluto.

Comprensión de MBR y GPT

Hay dos formas principales de almacenar información de partición en discos duros. El primero es MBR (*Master Boot Record*) y el segundo es GPT (*GUID Partition Table*).

MBR

Este es un remanente de los primeros días de MS-DOS (más específicamente, PC-DOS 2.0 de 1983) y durante décadas fue el esquema de particionamiento estándar en las PC. La tabla de particiones se almacena en el primer sector del disco, llamado *Boot Sector*, junto con un cargador de arranque, que en los sistemas Linux suele ser el *GRUB*. Pero MBR tiene una serie de limitaciones que dificultan su uso en sistemas modernos, como la imposibilidad de utilizar discos de más de 2 TB de tamaño y el límite de solo 4 particiones primarias por disco.

GUID

Un sistema de particiones que aborda muchas de las limitaciones de MBR. No existe un límite práctico en el tamaño del disco, y el número máximo de particiones está limitado solo por el propio sistema operativo. Se encuentra más comúnmente en máquinas más modernas que usan UEFI en lugar del antiguo BIOS de PC.

Durante las tareas de administración del sistema es muy posible que encuentre ambos esquemas en uso, por lo que es importante saber cómo usar las herramientas asociadas a cada uno para crear, eliminar o modificar particiones.

Gestión de particiones MBR con FDISK

La utilidad estándar para administrar particiones MBR en Linux es `fdisk`. Esta es una utilidad interactiva basada en menús. Para usarlo, teclee `fdisk` seguido del nombre del dispositivo correspondiente al disco que desea editar. Por ejemplo, el comando

```
# fdisk /dev/sda
```

editaría la tabla de particiones del primer dispositivo conectado a SATA (`sda`) en el sistema. Tenga en cuenta que debe especificar el dispositivo correspondiente al disco físico, no una de sus particiones (como `/dev/sda1`).

NOTE Todas las operaciones relacionadas con el disco en esta lección deben realizarse como el usuario `root` (el administrador del sistema), o con privilegios de root usando `sudo`.

Cuando se invoca, `fdisk` mostrará un saludo, luego una advertencia y esperará sus comandos.

```
# fdisk /dev/sda
Welcome to fdisk (util-linux 2.33.1).
Changes will remain in memory only, until you decide to write them.
Be careful before using the write command.

Command (m for help):
```

La advertencia es importante. Puede crear, editar o eliminar particiones a voluntad, pero no se escribirá nada en el disco a menos que utilice el comando write (`w`). Por lo tanto, puede "practicar" sin riesgo de perder datos, siempre que se mantenga alejado de la tecla `w`. Para salir de `fdisk` sin guardar los cambios, use el comando `q`.

NOTE Dicho esto, nunca debe practicar con un disco importante, ya que siempre existen riesgos. Utilice un disco externo de repuesto o una unidad flash USB.

Impresión de la tabla de particiones actual

El comando `p` se usa para imprimir la tabla de particiones actual. La salida sería algo como esto:

```
Command (m for help): p
Disk /dev/sda: 111.8 GiB, 120034123776 bytes, 234441648 sectors
Disk model: CT120BX500SSD1
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: dos
Disk identifier: 0x97f8fef5

Device     Boot     Start       End   Sectors   Size Id Type
/dev/sda1            4096 226048942 226044847 107.8G 83 Linux
/dev/sda2       226048944 234437550   8388607     4G 82 Linux swap / Solaris
```

Aquí está el significado de cada columna:

`Device`
: El dispositivo asignado a la partición.

`Boot`
: Muestra si la partición es “de arranque” o no.

`Start`
: El sector donde comienza la partición.

`End`
: El sector donde termina la partición.

`Sectors`
: El número total de sectores en la partición. Multiplíquelo por el tamaño del sector para obtener el tamaño de la partición en bytes.

`Size`
: El tamaño de la partición en formato “legible por humanos”. En el ejemplo anterior, los valores están en gigabytes.

`Id`
: El valor numérico que representa el tipo de partición.

`Type`
: La descripción del tipo de partición.

Particiones primarias vs extendidas

En un disco MBR, puede tener 2 tipos principales de particiones, *primarias* y *extendidas*. Como dijimos antes, solo puede tener 4 particiones primarias en el disco, y si desea que el disco sea “de arranque”, la primera partición debe ser primaria.

Una forma de evitar esta limitación es crear una partición extendida que actúe como contenedor de particiones *lógicas*. Podría tener, por ejemplo, una partición primaria, una partición extendida que ocupa el resto del espacio en disco y cinco particiones lógicas dentro de ella.

Para un sistema operativo como Linux, las particiones primarias y extendidas se tratan exactamente de la misma manera, por lo que no hay “ventajas” de usar una sobre la otra.

Crear una partición

Para crear una partición, use el comando `n`. De forma predeterminada, las particiones se crearán al comienzo del espacio no asignado en el disco. Se le pedirá el tipo de partición (primaria o extendida), primer sector y último sector.

Para el primer sector, normalmente puede aceptar el valor predeterminado sugerido por `fdisk`, a menos que necesite una partición para comenzar en un sector específico. En lugar de especificar el último sector, puede especificar un tamaño seguido de las letras `K`, `M`, `G`, `T` o `P` (Kilo, Mega, Giga, Tera o Peta). Por lo tanto, si desea crear una partición de 1 GB, puede especificar `+1G` como el `Último sector` y `fdisk` dimensionará la partición en consecuencia. Vea este ejemplo para la creación de una partición primaria:

```
Command (m for help): n
Partition type
   p   primary (0 primary, 0 extended, 4 free)
   e   extended (container for logical partitions)
Select (default p): p
Partition number (1-4, default 1): 1
First sector (2048-3903577, default 2048): 2048
Last sector, +/-sectors or +/-size{K,M,G,T,P} (2048-3903577, default 3903577): +1G
```

Comprobación de espacio no asignado

Si no sabe cuánto espacio libre hay en el disco, puede usar el comando `F` para mostrar el espacio no asignado, así:

```
Command (m for help): F
Unpartitioned space /dev/sdd: 881 MiB, 923841536 bytes, 1804378 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes

  Start     End Sectors  Size
2099200 3903577 1804378  881M
```

Eliminar particiones

Para eliminar una partición, use el comando `d`. `fdisk` le pedirá el número de la partición que desea eliminar, *a menos que* haya *sólo una* partición en el disco. En este caso, esta partición será *seleccionada y eliminada inmediatamente*.

Tenga en cuenta que si elimina una partición extendida, también se eliminarán todas las particiones lógicas que contiene.

¡Cuidado con la brecha!

Tenga en cuenta que al crear una nueva partición con `fdisk`, el tamaño máximo se limitará a la cantidad máxima de espacio *contiguo* no asignado en el disco. Digamos, por ejemplo, que tiene el siguiente mapa de particiones:

```
Device     Boot   Start     End Sectors  Size Id Type
/dev/sdd1          2048 1050623 1048576  512M 83 Linux
/dev/sdd2       1050624 2099199 1048576  512M 83 Linux
/dev/sdd3       2099200 3147775 1048576  512M 83 Linux
```

Luego borra la partición 2 y comprueba si hay espacio libre:

```
Command (m for help): F
Unpartitioned space /dev/sdd: 881 MiB, 923841536 bytes, 1804378 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes

  Start     End Sectors  Size
1050624 2099199 1048576  512M
3147776 3903577  755802  369M
```

Sumando el tamaño del espacio no asignado, en teoría tenemos 881 MB disponibles. Pero visualice lo que sucede cuando intentamos crear una partición de 700 MB:

```
Command (m for help): n
Partition type
   p   primary (2 primary, 0 extended, 2 free)
   e   extended (container for logical partitions)
Select (default p): p
Partition number (2,4, default 2): 2
First sector (1050624-3903577, default 1050624):
Last sector, +/-sectors or +/-size{K,M,G,T,P} (1050624-2099199, default 2099199):
+700M
Value out of range.
```

Eso sucede porque el espacio contiguo no asignado más grande en el disco es el bloque de 512 MB

que pertenecía a la partición 2. Su nueva partición no puede “pasar” la partición 3 para usar parte del espacio no asignado después de ella.

Cambiar el tipo de partición

En ocasiones, es posible que deba cambiar el tipo de partición, especialmente cuando se trata de discos que se utilizarán en otros sistemas operativos y plataformas. Esto se hace con el comando `t`, seguido del número de la partición que desea cambiar.

El tipo de partición debe ser especificado por su código hexadecimal correspondiente, y puede ver una lista de todos los códigos válidos usando el comando `l`.

No confunda el tipo de partición con el sistema de archivos que se utiliza en ella. Aunque al principio existía una relación entre ellos, hoy no se puede asumir que esto sea cierto. Una partición de Linux, por ejemplo, puede contener cualquier sistema de archivos nativo de Linux, como *ext4* o *ReiserFS*.

TIP Las particiones de Linux son del tipo `83` (Linux). Las particiones de intercambio son del tipo `82` (Linux Swap).

Administrar particiones GUID con GDISK

La utilidad `gdisk` es el equivalente a `fdisk` cuando se trata de discos particionados GPT. De hecho, la interfaz está modelada a partir de `fdisk`, con un indicador interactivo y los mismos (o muy similares) comandos.

Impresión de la tabla de particiones actual

El comando `p` se usa para imprimir la tabla de particiones actual. La salida sería algo como esto:

```
Command (? for help): p
Disk /dev/sdb: 3903578 sectors, 1.9 GiB
Model: DataTraveler 2.0
Sector size (logical/physical): 512/512 bytes
Disk identifier (GUID): AB41B5AA-A217-4D1E-8200-E062C54285BE
Partition table holds up to 128 entries
Main partition table begins at sector 2 and ends at sector 33
First usable sector is 34, last usable sector is 3903544
Partitions will be aligned on 2048-sector boundaries
Total free space is 1282071 sectors (626.0 MiB)

Number  Start (sector)    End (sector)  Size       Code  Name
   1            2048         2099199   1024.0 MiB  8300  Linux filesystem
   2         2623488         3147775   256.0 MiB   8300  Linux filesystem
```

Desde el principio, notamos algunas cosas diferentes:

- Cada disco tiene un identificador de disco (GUID) único. Este es un número hexadecimal de 128 bits, asignado al azar cuando se crea la tabla de particiones. Dado que hay 3.4×10^{38} valores posibles para este número, las posibilidades de que 2 discos aleatorios tengan el mismo GUID son bastante escasas. El GUID se puede usar para identificar qué sistemas de archivos montar en el momento del arranque (y dónde), eliminando la necesidad de usar la ruta del dispositivo para hacerlo (como `/dev/sdb`).
- ¿Ve la frase `Partition table holds up to 128 entries`? Así es, puede tener hasta 128 particiones en un disco GPT. Debido a esto, no hay necesidad de particiones *primarias* y *extendidas*.
- El espacio libre aparece en la última línea, por lo que no es necesario un equivalente del comando `F` de `fdisk`.

Crear una partición

El comando para crear una partición es `n`, al igual que en `fdisk`. La principal diferencia es que además del número de partición y el primer y último sector (o tamaño), también puede especificar el tipo de partición durante la creación. Las particiones GPT admiten muchos más tipos que MBR. Puede consultar una lista de todos los tipos admitidos mediante el comando `l`.

Eliminar particiones

Para eliminar una partición, escriba `d` y el número de partición. A diferencia de `fdisk`, la primera partición no se seleccionará automáticamente si es la única en el disco.

En los discos GPT, las particiones se pueden reordenar u "ordenar" fácilmente para evitar huecos en la secuencia de numeración. Para hacer esto, simplemente use el comando `s`. Por ejemplo, imagine un disco con la siguiente tabla de particiones:

```
Number  Start (sector)    End (sector)  Size       Code  Name
   1            2048         2099199   1024.0 MiB  8300  Linux filesystem
   2         2099200         2361343   128.0 MiB   8300  Linux filesystem
   3         2361344         2623487   128.0 MiB   8300  Linux filesystem
```

Si elimina la segunda partición, la tabla se convertiría en:

```
Number  Start (sector)    End (sector)  Size       Code  Name
   1            2048         2099199   1024.0 MiB  8300  Linux filesystem
   3         2361344         2623487   128.0 MiB   8300  Linux filesystem
```

Si usa el comando `s`, se convertiría en:

```
Number  Start (sector)    End (sector)  Size       Code  Name
   1            2048         2099199   1024.0 MiB  8300  Linux filesystem
   2         2361344         2623487   128.0 MiB   8300  Linux filesystem
```

Observe que la tercera partición se convirtió en la segunda.

¿Brecha? ¿Qué brecha?

A diferencia de los discos MBR, al crear una partición en discos GPT, el tamaño no está limitado por la cantidad máxima de espacio *contiguo* no asignado. Puede utilizar hasta el último bit de un sector libre, sin importar dónde se encuentre en el disco.

Opciones de recuperación

Los discos GPT almacenan copias de seguridad del encabezado GPT y la tabla de particiones, lo que facilita la recuperación de discos en caso de que estos datos se hayan dañado. `gdisk` proporciona funciones para ayudar en esas tareas de recuperación, a las que se accede con el comando `r`.

Puede reconstruir un encabezado GPT principal corrupto o una tabla de particiones con `b` y `c`, respectivamente, o usar el encabezado principal y la tabla para reconstruir una copia de seguridad con `d` y `e`. También puede convertir un MBR a GPT con `f` y hacer lo contrario con `g`, entre otras operaciones. Escriba `?` En el menú de recuperación para obtener una lista de todos los comandos de

recuperación disponibles y descripciones sobre lo que hacen.

Creación de sistemas de archivos

Particionar el disco es solo el primer paso para poder usar un disco. Después de eso, deberá formatear la partición con un sistema de archivos antes de usarla para almacenar datos.

Un sistema de archivos controla cómo se almacenan los datos y cómo se accede a ellos en el disco. Linux admite muchos sistemas de archivos, algunos nativos, como la familia ext (Extended Filesystem), mientras que otros provienen de otros sistemas operativos como FAT de MS-DOS, NTFS de Windows NT, HFS y HFS+ de Mac OS, etc.

La herramienta estándar utilizada para crear un sistema de archivos en Linux es `mkfs`, que viene en muchos “sabores” según el sistema de archivos con el que necesita trabajar.

Creación de un sistema de archivos ext2/ext3/ext4

El *Sistema de archivos extendido* (ext) fue el primer sistema de archivos para Linux, y a través de los años fue reemplazado por nuevas versiones llamadas ext2, ext3 y ext4, que actualmente es el sistema de archivos predeterminado para muchas distribuciones de Linux.

Las utilidades `mkfs.ext2`, `mkfs.ext3` y `mkfs.ext4` se utilizan para crear sistemas de archivos ext2, ext3 y ext4 respectivamente. De hecho, todas estas “utilidades” existen sólo como enlaces simbólicos a otra utilidad llamada `mke2fs`. `mke2fs` modifica sus valores predeterminados de acuerdo con el nombre por el que se le llama. Como tal, todos tienen el mismo comportamiento y parámetros de línea de comando.

La forma de uso más sencilla es:

```
# mkfs.ext2 TARGET
```

Donde `TARGET` es el nombre de la partición donde se debe crear el sistema de archivos. Por ejemplo, para crear un sistema de archivos ext3 en `/dev/sdb1` el comando sería:

```
# mkfs.ext3 /dev/sdb1
```

En lugar de usar el comando correspondiente al sistema de archivos que desea crear, puede pasar el parámetro `-t` a `mke2fs` seguido del nombre del sistema de archivos. Por ejemplo, los siguientes comandos son equivalentes y crearán un sistema de archivos ext4 en `/dev/sdb1`.

```
# mkfs.ext4 /dev/sdb1
# mke2fs -t ext4 /dev/sdb1
```

Parámetros de línea de comandos

`mke2fs` admite una amplia gama de opciones y parámetros de línea de comandos. Éstos son algunos de los más importantes. Todos ellos también se aplican a `mkfs.ext2`, `mkfs.ext3` y `mkfs.ext4`:

`-b SIZE`

Establece el tamaño de los bloques de datos en el dispositivo en `SIZE`, que puede ser de 1024, 2048 o 4096 bytes por bloque.

`-c`

Comprueba el dispositivo de destino en busca de bloques defectuosos antes de crear el sistema de archivos. Puede ejecutar una comprobación completa, pero mucho más lenta, pasando este parámetro dos veces, como en `mkfs.ext4 -c -c TARGET`.

`-d DIRECTORY`

Copia el contenido del directorio especificado en la raíz del nuevo sistema de archivos. Útil si necesita “rellenar previamente” el disco con un conjunto predefinido de archivos.

`-F`

¡Peligro, Will Robinson! Esta opción *forzará* mke2fs a crear un sistema de archivos, incluso si las otras opciones pasadas a él o al objetivo son peligrosas o no tienen ningún sentido. Si se especifica dos veces (como en `-F -F`), incluso se puede usar para crear un sistema de archivos en un dispositivo que está montado o en uso, lo cual es muy, *muy* malo.

`-L VOLUME_LABEL`

Establecerá la etiqueta de volumen en la especificada en `VOLUME_LABEL`. Esta etiqueta debe tener un máximo de 16 caracteres.

`-n`

Esta es una opción realmente útil que simula la creación del sistema de archivos y muestra lo que se haría si se ejecutara sin la opción `n`. Piense en ello como un modo de “prueba”. Es bueno comprobar las cosas antes de realizar cambios en el disco.

`-q`

Modo silencioso. `mke2fs` se ejecutará normalmente, pero no producirá ninguna salida en la terminal. Útil cuando se ejecuta `mke2fs` desde un script.

`-U ID`

Esto establecerá el UUID (*Universally Unique Identifier*) de una partición en el valor especificado como ID. Los UUID son números de 128 bits en notación hexadecimal que sirven para identificar de forma única una partición en el sistema operativo. Este número se especifica como una cadena de 32 dígitos en el formato 8-4-4-4-12, es decir, 8 dígitos, guión, 4 dígitos, guión, 4 dígitos, guión, 4 dígitos, guión, 12 dígitos, como `D249E380-7719-45A1-813C-35186883987E`. En lugar de un ID, también puede especificar parámetros como `clear` para borrar el UUID del sistema de archivos, `random`, para usar un UUID generado aleatoriamente o `time` para crear un UUID basado en el tiempo.

`-V`

Modo detallado, imprime mucha más información durante el funcionamiento de lo habitual. Útil para fines de depuración.

Creación de un sistema de archivos XFS

XFS es un sistema de archivos de alto rendimiento desarrollado originalmente por Silicon Graphics en 1993 para su sistema operativo IRIX. Debido a sus características de rendimiento y confiabilidad, se usa comúnmente para servidores y otros entornos que requieren un ancho de banda alto (o garantizado) del sistema de archivos.

Las herramientas para administrar sistemas de archivos XFS son parte del paquete `xfsprogs`. Es posible que este paquete deba instalarse manualmente, ya que no se incluye de forma predeterminada en algunas distribuciones de Linux. Otros, como Red Hat Enterprise Linux 7, usan XFS como sistema de archivos predeterminado.

Los sistemas de archivos XFS se dividen en al menos 2 partes, una *sección de registro* donde se mantiene un registro de todas las operaciones del sistema de archivos (comúnmente llamado *Journal*) y la *sección de datos*. La sección de registro puede estar ubicada dentro de la sección de datos (el comportamiento predeterminado), o incluso en un disco separado por completo, para un mejor rendimiento y confiabilidad.

El comando más básico para crear un sistema de archivos XFS es `mkfs.xfs TARGET`, donde `TARGET` es la partición en la que desea que se cree el sistema de archivos. Por ejemplo: `mkfs.xfs /dev/sda1`.

Como en `mke2fs`, `mkfs.xfs` admite una serie de opciones de línea de comandos. Éstos son algunas de las más comunes.

`-b size=VALUE`

Establece el tamaño de bloque en el sistema de archivos, en bytes, al especificado en `VALUE`. El valor predeterminado es 4096 bytes (4 KiB), el mínimo es 512 y el máximo es 65536 (64 KiB).

`-m crc=VALUE`

Los parámetros que comienzan con `-m` son opciones de metadatos. Éste habilita (si `VALUE` es `1`) o deshabilita (si `VALUE` es `0`) el uso de comprobaciones CRC32c para verificar la integridad de todos los metadatos en el disco. Esto permite una mejor detección de errores y recuperación de fallas relacionadas con problemas de hardware, por lo que está habilitado de forma predeterminada. El impacto en el rendimiento de esta comprobación debería ser mínimo, por lo que normalmente no hay razón para desactivarlo.

`-m uuid=VALUE`

Establece el UUID de la partición al especificado como VALUE. Recuerde que los UUID son números de 32 caracteres (128 bits) en base hexadecimal, especificados en grupos de 8, 4, 4, 4 y 12 dígitos separados por guiones, como `1E83E3A3-3AE9-4AAC-BF7E-29DFFECD36C0`.

`-f`

Forzar la creación de un sistema de archivos en el dispositivo de destino incluso si se detecta un sistema de archivos en él.

`-l logdev=DEVICE`

Esto colocará la sección de registro del sistema de archivos en el dispositivo especificado, en lugar de dentro de la sección de datos.

`-l size=VALUE`

Esto establecerá el tamaño de la sección de registro en el especificado en `VALUE`. El tamaño se puede especificar en bytes y se pueden utilizar sufijos como `m` o `g`. `-l size=10m`, por ejemplo, limitará la sección de registro a 10 Megabytes.

`-q`

Modo silencioso. En este modo, `mkfs.xfs` no imprimirá los parámetros del sistema de archivos que se está creando.

`-L LABEL`

Establece la etiqueta del sistema de archivos, que puede tener un máximo de 12 caracteres.

`-N`

Similar al parámetro `-n` de `mke2fs`, hará que `mkfs.xfs` imprima todos los parámetros para la creación del sistema de archivos, sin crearlo realmente.

Creación de un sistema de archivos FAT o VFAT

El sistema de archivos FAT se originó a partir de MS-DOS y, a lo largo de los años, ha recibido

muchas revisiones que culminaron con el formato FAT32 lanzado en 1996 con Windows 95 OSR2.

VFAT es una extensión del formato FAT16 que admite nombres de archivo largos (hasta 255 caracteres). Ambos sistemas de archivos son manejados por la misma utilidad, `mkfs.fat`. `mkfs.vfat` es un alias.

El sistema de archivos FAT tiene importantes inconvenientes que restringen su uso en discos grandes. FAT16, por ejemplo, admite volúmenes de 4 GB como máximo y un tamaño de archivo máximo de 2 GB. FAT32 aumenta el tamaño del volumen hasta 2 PB y el tamaño máximo del archivo hasta 4 GB. Debido a esto, los sistemas de archivos FAT se utilizan hoy con más frecuencia en unidades flash pequeñas o tarjetas de memoria (de hasta 2 GB de tamaño), o en dispositivos y sistemas operativos heredados que no admiten sistemas de archivos más avanzados.

El comando más básico para la creación de un sistema de archivos FAT es `mkfs.fat TARGET`, donde `TARGET` es la partición en la que desea que se cree el sistema de archivos. Por ejemplo: `mkfs.fat /dev/sdc1`.

Como otras utilidades, `mkfs.fat` soporta una serie de opciones de línea de comandos. A continuación se muestran los más importantes. Se puede leer una lista completa y una descripción de cada opción en el manual de la utilidad, con el comando `man mkfs.fat`.

`-c`

Comprueba el dispositivo de destino en busca de bloques defectuosos antes de crear el sistema de archivos.

`-C FILENAME BLOCK_COUNT`

Creará el archivo especificado en `FILENAME` y luego creará un sistema de archivos FAT dentro de él, creando efectivamente una “imagen de disco” vacía, que luego puede escribirse en un dispositivo usando una utilidad como `dd` o montarse como un loopback dispositivo. Al usar esta opción, el número de bloques en el sistema de archivos (`BLOCK_COUNT`) debe especificarse después del nombre del dispositivo.

`-F SIZE`

Selecciona el tamaño de la FAT (*File Allocation Table*), entre 12, 16 o 32, es decir, entre FAT12, FAT16 o FAT32. Si no se especifica, `mkfs.fat` seleccionará la opción apropiada según el tamaño del sistema de archivos.

`-n NAME`

Establece la etiqueta de volumen, o el nombre, para el sistema de archivos. Puede tener hasta 11 caracteres y el valor predeterminado es sin nombre.

`-v`
: Modo detallado. Imprime mucha más información de la habitual, útil para depurar.

> **NOTE** `mkfs.fat` *no puede* crear un sistema de archivos “de arranque”. Según la página del manual, “esto no es tan fácil como podría pensar” y no se implementará.

Creación de un sistema de archivos exFAT

exFAT es un sistema de archivos creado por Microsoft en 2006 que aborda una de las limitaciones más importantes de FAT32: el tamaño del archivo y del disco. En exFAT, el tamaño máximo de archivo es de 16 exabytes (desde 4 GB en FAT32) y el tamaño máximo del disco es de 128 petabytes.

Como es compatible con los tres principales sistemas operativos (Windows, Linux y mac OS), es una buena opción donde se necesita interoperabilidad, como en unidades flash de gran capacidad, tarjetas de memoria y discos externos. De hecho, es el sistema de archivos predeterminado, según lo define la *SD Association*, para tarjetas de memoria SDXC de más de 32 GB.

La utilidad predeterminada para crear sistemas de archivos exFAT es `mkfs.exfat`, que es un enlace a `mkexfatfs`. El comando más básico es `mkfs.exfat TARGET`, donde `TARGET` es la partición en la que desea que se cree el sistema de archivos. Por ejemplo: `mkfs.exfat /dev/sdb2`.

Al contrario de las otras utilidades discutidas en esta lección, `mkfs.exfat` tiene muy pocas opciones de línea de comandos. Son:

`-i VOL_ID`
: Establece el ID de volumen en el valor especificado en `VOL_ID`. Este es un número hexadecimal de 32 bits. Si no está definido, se establece una ID basada en la hora actual.

`-n NAME`
: Establece la etiqueta de volumen o el nombre. Puede tener hasta 15 caracteres y el valor predeterminado es sin nombre.

`-p SECTOR`
: Especifica el primer sector de la primera partición del disco. Este es un valor opcional y el predeterminado es cero.

`-s SECTORS`
: Define el número de sectores físicos por grupo de asignación. Debe ser una potencia de dos, como 1, 2, 4, 8, etc.

Familiarización con el sistema de archivos Btrfs

Btrfs (oficialmente el *B-Tree Filesystem*, pronunciado como “Butter FS”, “Better FS” o incluso “Butterfuss”, su elección) es un sistema de archivos que ha estado en desarrollo desde 2007 específicamente para Linux por el Oracle Corporation y otras empresas, incluidas Fujitsu, Red Hat, Intel y SUSE, entre otras.

Hay muchas características que hacen que Btrfs sea atractivo en sistemas modernos donde son comunes grandes cantidades de almacenamiento. Entre estos se encuentran el soporte para múltiples dispositivos (incluyendo la creación de bandas, la duplicación y la creación de bandas+duplicación, como en una configuración RAID), compresión transparente, optimizaciones de SSD, copias de seguridad incrementales, instantáneas, desfragmentación en línea, comprobaciones fuera de línea, compatibilidad con subvolúmenes (con cuotas), deduplicación y mucho más.

Como es un sistema de archivos *copy-on-write*, es muy resistente a los bloqueos. Y además de eso, Btrfs es fácil de usar y está bien soportado por muchas distribuciones de Linux. Y algunos de ellos, como SUSE, lo utilizan como sistema de archivos predeterminado.

NOTE En un sistema de archivos tradicional, cuando desea sobrescribir parte de un archivo, los datos nuevos se colocan directamente sobre los datos antiguos que están reemplazando. En un sistema de archivos *copy-on-write*, los nuevos datos se escriben en el espacio libre en el disco, luego los metadatos originales del archivo se actualizan para hacer referencia a los nuevos datos y solo entonces se liberan los datos antiguos, porque ya no son necesarios. Esto reduce las posibilidades de pérdida de datos en caso de una falla, ya que los datos antiguos solo se descartan después de que el sistema de archivos esté absolutamente seguro de que ya no se necesitan y los nuevos datos están en su lugar.

Creación de un sistema de archivos Btrfs

La utilidad `mkfs.btrfs` se utiliza para crear un sistema de archivos Btrfs. Usar el comando sin ninguna opción crea un sistema de archivos Btrfs en un dispositivo dado, así:

```
# mkfs.btrfs /dev/sdb1
```

TIP Si no tiene la utilidad `mkfs.btrfs` en su sistema, busque `btrfs-progs` en el administrador de paquetes de su distribución.

Puede usar la `-L` para establecer una etiqueta (o nombre) para su sistema de archivos. Las etiquetas Btrfs pueden tener hasta 256 caracteres, excepto para las nuevas líneas:

```
# mkfs.btrfs /dev/sdb1 -L "New Disk"
```

TIP Incluya la etiqueta entre comillas (como arriba) si contiene espacios.

Tenga en cuenta este aspecto peculiar de Btrfs: puede pasar *múltiples* dispositivos al comando `mkfs.btrfs`. Pasar más de un dispositivo abarcará el sistema de archivos sobre todos los dispositivos, lo que es similar a una configuración RAID o LVM. Para especificar cómo se distribuirán los metadatos en la matriz de discos, use el parámetro `-m`. Los parámetros válidos son `raid0`, `raid1`, `raid5`, `raid6`, `raid10`, `single` y `dup`.

Por ejemplo, para crear un sistema de archivos que abarque `/dev/sdb1` y `/dev/sdc1`, concatenando las dos particiones en una gran partición, use:

```
# mkfs.btrfs -d single -m single /dev/sdb /dev/sdc
```

WARNING Los sistemas de archivos que abarcan varias particiones, como el anterior, pueden parecer ventajosos al principio, pero no son una buena idea desde el punto de vista de la seguridad de los datos, ya que una falla en un solo disco de la matriz significa cierta pérdida de datos. El riesgo aumenta a medida que utiliza más discos, ya que también tiene más puntos posibles de falla.

Gestión de subvolúmenes

Los subvolúmenes son como sistemas de archivos dentro de sistemas de archivos. Piense en ellos como un directorio que se puede montar (y tratar como) un sistema de archivos separado. Los subvolúmenes facilitan la organización y la administración del sistema, ya que cada uno de ellos puede tener cuotas o reglas instantáneas independientes.

NOTE Los subvolúmenes no son particiones. Una partición asigna un espacio fijo en una unidad. Esto puede generar problemas más adelante, como que una partición se quede sin espacio cuando a otra le queda mucho espacio. No es así con los subvolúmenes, ya que “comparten” el espacio libre de su sistema de archivos raíz y crecen según sea necesario.

Suponga que tiene un sistema de archivos Btrfs montado en `/mnt/disk`, y desea crear un subvolumen dentro de él para almacenar sus copias de seguridad. Llamémoslo `BKP`:

```
# btrfs subvolume create /mnt/disk/BKP
```

A continuación, enumeramos el contenido del sistema de archivos `/mnt/disk`. Verá que tenemos un nuevo directorio, llamado así por nuestro subvolumen.

```
$ ls -lh /mnt/disk/
total 0
drwxr-xr-x 1 root   root     0 jul 13 17:35 BKP
drwxrwxr-x 1 carol carol 988 jul 13 17:30 Images
```

NOTE Sí, también se puede acceder a los subvolúmenes como a cualquier otro directorio.

Podemos comprobar que el subvolumen está activo, con el comando:

```
# btrfs subvolume show /mnt/disk/BKP/
    Name:            BKP
    UUID:            e90a1afe-69fa-da4f-9764-3384f66fa32e
    Parent UUID:         -
    Received UUID:       -
    Creation time:       2019-07-13 17:35:40 -0300
    Subvolume ID:        260
    Generation:          23
    Gen at creation:     22
    Parent ID:       5
    Top level ID:        5
    Flags:           -
    Snapshot(s):
```

Puede montar el subvolumen en `/mnt/BKP` pasando el parámetro `-t btrfs -o subvol=NAME` al comando `mount`:

```
# mount -t btrfs -o subvol=BKP /dev/sdb1 /mnt/bkp
```

NOTE El parámetro `-t` especifica el tipo de sistema de archivos que se va a montar.

Trabajar con instantáneas

Las instantáneas son como subvolúmenes, pero rellenadas previamente con el contenido del volumen desde el que se tomó la instantánea.

Cuando se crea, una instantánea y el volumen original tienen exactamente el mismo contenido. Pero a partir de ese momento, divergirán. Los cambios realizados en el volumen original (como archivos agregados, renombrados o eliminados) no se reflejarán en la instantánea y viceversa.

Tenga en cuenta que una instantánea *no* duplica los archivos e inicialmente casi no ocupa espacio en el disco. Simplemente duplica el árbol del sistema de archivos, mientras apunta a los datos originales.

El comando para crear una instantánea es el mismo que se usa para crear un subvolumen, simplemente agregue el parámetro `snapshot` después de `btrfs subvolume`. El siguiente comando creará una instantánea del sistema de archivos Btrfs montado en `/mnt/disk` en `/mnt/disk/snap`:

```
# btrfs subvolume snapshot /mnt/disk /mnt/disk/snap
```

Ahora, imagine que tiene el siguiente contenido en `/mnt/disk`:

```
$ ls -lh
total 2,8M
-rw-rw-r-- 1 carol carol 109K jul 10 16:22 Galaxy_Note_10.png
-rw-rw-r-- 1 carol carol 484K jul  5 15:01 geminoid2.jpg
-rw-rw-r-- 1 carol carol 429K jul  5 14:52 geminoid.jpg
-rw-rw-r-- 1 carol carol 467K jul  2 11:48 LG-G8S-ThinQ-Mirror-White.jpg
-rw-rw-r-- 1 carol carol 654K jul  2 11:39 LG-G8S-ThinQ-Range.jpg
-rw-rw-r-- 1 carol carol  94K jul  2 15:43 Mimoji_Comparativo.jpg
-rw-rw-r-- 1 carol carol 112K jul 10 16:20 Note10Plus.jpg
drwx------ 1 carol carol  366 jul 13 17:56 snap
-rw-rw-r-- 1 carol carol 118K jul 11 16:36 Twitter_Down_20190711.jpg
-rw-rw-r-- 1 carol carol 324K jul  2 15:22 Xiaomi_Mimoji.png
```

Observe el directorio de instantáneas, que contiene la instantánea. Ahora eliminemos algunos archivos y verifiquemos el contenido del directorio:

```
$ rm LG-G8S-ThinQ-*
$ ls -lh
total 1,7M
-rw-rw-r-- 1 carol carol 109K jul 10 16:22 Galaxy_Note_10.png
-rw-rw-r-- 1 carol carol 484K jul  5 15:01 geminoid2.jpg
-rw-rw-r-- 1 carol carol 429K jul  5 14:52 geminoid.jpg
-rw-rw-r-- 1 carol carol  94K jul  2 15:43 Mimoji_Comparativo.jpg
-rw-rw-r-- 1 carol carol 112K jul 10 16:20 Note10Plus.jpg
drwx------ 1 carol carol  366 jul 13 17:56 snap
-rw-rw-r-- 1 carol carol 118K jul 11 16:36 Twitter_Down_20190711.jpg
-rw-rw-r-- 1 carol carol 324K jul  2 15:22 Xiaomi_Mimoji.png
```

Sin embargo, si revisa dentro del directorio snap, los archivos que eliminó todavía están allí y se pueden restaurar si es necesario.

```
$ ls -lh snap/
total 2,8M
-rw-rw-r-- 1 carol carol 109K jul 10 16:22 Galaxy_Note_10.png
-rw-rw-r-- 1 carol carol 484K jul  5 15:01 geminoid2.jpg
-rw-rw-r-- 1 carol carol 429K jul  5 14:52 geminoid.jpg
-rw-rw-r-- 1 carol carol 467K jul  2 11:48 LG-G8S-ThinQ-Mirror-White.jpg
-rw-rw-r-- 1 carol carol 654K jul  2 11:39 LG-G8S-ThinQ-Range.jpg
-rw-rw-r-- 1 carol carol  94K jul  2 15:43 Mimoji_Comparativo.jpg
-rw-rw-r-- 1 carol carol 112K jul 10 16:20 Note10Plus.jpg
-rw-rw-r-- 1 carol carol 118K jul 11 16:36 Twitter_Down_20190711.jpg
-rw-rw-r-- 1 carol carol 324K jul  2 15:22 Xiaomi_Mimoji.png
```

También es posible crear instantáneas de solo lectura. Funcionan exactamente como instantáneas en las que se puede escribir, con la diferencia de que el contenido de la instantánea no se puede cambiar, se “congelan” en el tiempo. Simplemente agregue el parámetro `-r` al crear la instantánea:

```
# btrfs subvolume snapshot -r /mnt/disk /mnt/disk/snap
```

Algunas palabras sobre compresión

Btrfs admite la compresión de archivos transparente, con tres algoritmos diferentes disponibles para el usuario. Esto se hace automáticamente por archivo, siempre que el sistema de archivos esté montado con la opción `-o compress`. Los algoritmos son lo suficientemente inteligentes como para detectar archivos que no se pueden comprimir y no intentarán comprimirlos, ahorrando recursos del sistema. Entonces, en un solo directorio puede tener archivos comprimidos y descomprimidos juntos.

El algoritmo de compresión predeterminado es ZLIB, pero están disponibles LZO (más rápido, peor relación de compresión) o ZSTD (más rápido que ZLIB, compresión comparable), con múltiples niveles de compresión (consulte el objetivo correspondiente en las opciones de montaje).

Administrar particiones con GNU Parted

GNU Parted es un editor de particiones muy poderoso (de ahí el nombre) que se puede usar para crear, eliminar, mover, redimensionar, rescatar y copiar particiones. Puede funcionar con discos GPT y MBR y cubrir casi todas sus necesidades de administración de discos.

Hay muchas interfaces gráficas que facilitan mucho el trabajo con `parted`, como *GParted* para entornos de escritorio basados en GNOME y el *KDE Partition Manager* para escritorios KDE. Sin embargo, debe aprender a usar `parted` en la línea de comandos, ya que en una configuración de servidor nunca puede contar con un entorno de escritorio gráfico disponible.

WARNING

A diferencia de `fdisk` y `gdisk`, parted realiza cambios en el disco *inmediatamente* después de que se emite el comando, sin esperar a que otro comando escriba los cambios en el disco. Al practicar, es aconsejable hacerlo en un disco o unidad flash vacíos o de repuesto, para que no exista riesgo de pérdida de datos en caso de que cometa un error.

La forma más sencilla de comenzar a usar parted es escribiendo `parted DEVICE`, donde `DEVICE` es el dispositivo que desea administrar (`parted /dev/sdb`). El programa inicia una interfaz de línea de comandos interactiva como `fdisk` y `gdisk` con un mensaje `(parted)` para que ingrese los comandos.

```
# parted /dev/sdb
GNU Parted 3.2
Using /dev/sdb
Welcome to GNU Parted! Type 'help' to view a list of commands.

(parted)
```

WARNING

¡Tenga cuidado! Si no especifica un dispositivo, `parted` seleccionará automáticamente el disco principal (normalmente `/dev/sda`) para trabajar.

Seleccionar discos

Para cambiar a un disco diferente al especificado en la línea de comando, puede usar el comando `select`, seguido del nombre del dispositivo:

```
(parted) select /dev/sdb
Using /dev/sdb
```

Obtener información

El comando `print` se puede utilizar para obtener más información sobre una partición específica o incluso todos los dispositivos de bloque (discos) conectados a su sistema.

Para obtener información sobre la partición seleccionada actualmente, simplemente teclee `print`:

```
(parted) print
Model: ATA CT120BX500SSD1 (scsi)
Disk /dev/sda: 120GB
Sector size (logical/physical): 512B/512B
Partition Table: msdos
Disk Flags:

Number  Start   End    Size    Type     File system     Flags
 1      2097kB  116GB  116GB   primary  ext4
 2      116GB   120GB  4295MB  primary  linux-swap(v1)
```

Puede obtener una lista de todos los dispositivos de bloque conectados a su sistema usando `print devices`:

```
(parted) print devices
/dev/sdb (1999MB)
/dev/sda (120GB)
/dev/sdc (320GB)
/dev/mapper/cryptswap (4294MB)
```

Para obtener información sobre todos los dispositivos conectados a la vez, puede usar `print all`. Si desea saber cuánto espacio libre hay en cada uno de ellos, puede usar `print free`:

```
(parted) print free
Model: ATA CT120BX500SSD1 (scsi)
Disk /dev/sda: 120GB
Sector size (logical/physical): 512B/512B
Partition Table: msdos
Disk Flags:

Number  Start   End     Size    Type     File system     Flags
        32.3kB  2097kB  2065kB           Free Space
 1      2097kB  116GB   116GB   primary  ext4
        116GB   116GB   512B             Free Space
 2      116GB   120GB   4295MB  primary  linux-swap(v1)
        120GB   120GB   2098kB           Free Space
```

Crear una tabla de particiones en un disco vacío

Para crear una tabla de particiones en un disco vacío, use el comando `mklabel`, seguido del tipo de tabla de particiones que desea usar.

Hay muchos tipos de tablas de particiones admitidas, pero los tipos principales que debe conocer son `msdos` que se usa aquí para referirse a una tabla de particiones MBR, y `gpt` para referirse a una tabla de particiones GPT. Para crear una tabla de particiones MBR, teclee:

```
(parted) mklabel msdos
```

Y para crear una tabla de particiones GPT, el comando es:

```
(parted) mklabel gpt
```

Crear una partición

Para crear una partición se usa el comando `mkpart`, usando la sintaxis `mkpart PARTTYPE FSTYPE START END`, donde:

`PARTTYPE`

Es el tipo de partición, que puede ser `primaria`, `lógica` o `extendida` en caso de que se utilice una tabla de particiones MBR.

FSTYPE

Especifica qué sistema de archivos se utilizará en esta partición. Tenga en cuenta que `parted` *no* creará el sistema de archivos. Simplemente establece una marca en la partición que le dice al sistema operativo qué tipo de datos esperar de ella.

START

Especifica el punto exacto en el dispositivo donde comienza la partición. Puede utilizar diferentes unidades para especificar este punto. `2s` se puede usar para referirse al segundo sector del disco, mientras que `1m` se refiere al comienzo del primer megabyte del disco. Otras unidades comunes son `B` (bytes) y `%` (porcentaje del disco).

END

Especifica el final de la partición. Tenga en cuenta que este *no* es el tamaño de la partición, es *el punto del disco donde termina*. Por ejemplo, si especifica `100 m`, la partición finalizará 100 MB después del inicio del disco. Puede utilizar las mismas unidades que en el parámetro `START`.

Entonces, el comando:

```
(parted) mkpart primary ext4 1m 100m
```

Crea una partición primaria de tipo `ext4`, comenzando en el primer megabyte del disco y terminando después del megabyte 100.

Eliminar una partición

Para eliminar una partición, use el comando `rm` seguido del número de partición, que puede mostrar usando el comando `print`. Entonces, `rm 2` eliminaría la segunda partición en el disco seleccionado actualmente.

Recuperando particiones

`parted` puede recuperar una partición eliminada. Considere que tiene la siguiente estructura de partición:

```
Number  Start   End     Size    File system  Name     Flags
 1      1049kB  99.6MB  98.6MB  ext4         primary
 2      99.6MB  200MB   100MB   ext4         primary
 3      200MB   300MB   99.6MB  ext4         primary
```

Por accidente, eliminó la partición 2 usando `rm 2`. Para recuperarlo, puede utilizar el comando `rescue`, con la sintaxis `rescue START END`, donde `START` es la ubicación aproximada donde comenzó la partición y `END` la ubicación aproximada donde terminó.

`parted` escaneará el disco en busca de particiones y ofrecerá restaurar las que se encuentren. En el ejemplo anterior, la partición `2` comenzó en 99,6 MB y terminó en 200 MB. Entonces puede usar el siguiente comando para recuperar la partición:

```
(parted) rescue 90m 210m
Information: A ext4 primary partition was found at 99.6MB -> 200MB.
Do you want to add it to the partition table?

Yes/No/Cancel? y
```

Esto recuperará la partición y su contenido. Tenga en cuenta que `rescue` solo puede recuperar particiones que tengan un sistema de archivos instalado. No se detectan particiones vacías.

Cambiar el tamaño de las particiones ext2/3/4

`parted` se puede usar para cambiar el tamaño de las particiones y hacerlas más grandes o más pequeñas. Sin embargo, hay algunas advertencias:

- Durante el cambio de tamaño, la partición debe estar sin usar y sin montar.
- Necesita suficiente espacio libre *después* de la partición para hacerla crecer al tamaño que desee.

El comando es `resizepart`, seguido del número de partición y dónde debería terminar. Por ejemplo, si tiene la siguiente tabla de particiones:

```
Number  Start   End     Size    File system  Name     Flags
 1      1049kB  99.6MB  98.6MB  ext4         primary
 2      99.6MB  200MB   100MB   ext4
 3      200MB   300MB   99.6MB  ext4         primary
```

Intentar hacer crecer la partición `1` usando `resizepart` generaría un mensaje de error, porque con el nuevo tamaño, la partición `1` se superpondría con la partición `2`. Sin embargo, la partición `3` se puede cambiar de tamaño ya que hay espacio libre después de ella, lo que se puede verificar con el comando `print free`:

```
(parted) print free
Model: Kingston DataTraveler 2.0 (scsi)
Disk /dev/sdb: 1999MB
Sector size (logical/physical): 512B/512B
Partition Table: gpt
Disk Flags:

 Number  Start   End     Size    File system  Name     Flags
        17.4kB  1049kB  1031kB  Free Space
 1      1049kB  99.6MB  98.6MB  ext4         primary
 2      99.6MB  200MB   100MB   ext4
 3      200MB   300MB   99.6MB  ext4         primary
        300MB   1999MB  1699MB  Free Space
```

Entonces puede usar el siguiente comando para cambiar el tamaño de la partición 3 a 350 MB:

```
(parted) resizepart 3 350m

(parted) print
Model: Kingston DataTraveler 2.0 (scsi)
Disk /dev/sdb: 1999MB
Sector size (logical/physical): 512B/512B
Partition Table: gpt
Disk Flags:

Number  Start   End     Size    File system  Name     Flags
 1      1049kB  99.6MB  98.6MB  ext4         primary
 2      99.6MB  200MB   100MB   ext4
 3      200MB   350MB   150MB   ext4         primary
```

Recuerde que el nuevo punto final se especifica contando desde el inicio del disco. Entonces, debido a que la partición `3` terminó en 300 MB, ahora debe terminar en 350 MB.

Pero cambiar el tamaño de la partición es solo una parte de la tarea. También necesita cambiar el tamaño del sistema de archivos que reside en ella. Para sistemas de archivos ext2/3/4 esto se hace con el comando `resize2fs`. En el caso del ejemplo anterior, la partición 3 todavía muestra el tamaño "antiguo" cuando se monta:

```
$ df -h /dev/sdb3
Filesystem      Size  Used Avail Use% Mounted on
/dev/sdb3        88M  1.6M   80M   2% /media/carol/part3
```

Para ajustar el tamaño, se puede usar el comando `resize2fs DEVICE SIZE`, donde `DEVICE` corresponde a la partición que desea cambiar de tamaño y `SIZE` es el nuevo tamaño. Si omite el parámetro de tamaño, utilizará todo el espacio disponible de la partición. Antes de cambiar el tamaño, se recomienda desmontar la partición.

En el ejemplo anterior:

```
$ sudo resize2fs /dev/sdb3
resize2fs 1.44.6 (5-Mar-2019)
Resizing the filesystem on /dev/sdb3 to 146212 (1k) blocks.
The filesystem on /dev/sdb3 is now 146212 (1k) blocks long.

$ df -h /dev/sdb3
Filesystem      Size  Used Avail Use% Mounted on
/dev/sdb3       135M  1.6M  123M   2% /media/carol/part3
```

Para *encoger* una partición, el proceso debe realizarse en orden inverso. *Primero* cambie el tamaño del sistema de archivos al nuevo tamaño más pequeño, luego cambie el tamaño de la partición usando `parted`.

WARNING Preste atención al encoger particiones. Si ordena mal las cosas, ¡perderá datos!

En nuestro ejemplo:

```
# resize2fs /dev/sdb3 88m
resize2fs 1.44.6 (5-Mar-2019)
Resizing the filesystem on /dev/sdb3 to 90112 (1k) blocks.
The filesystem on /dev/sdb3 is now 90112 (1k) blocks long.

# parted /dev/sdb3
(parted) resizepart 3 300m
Warning: Shrinking a partition can cause data loss, are you sure
you want to continue?

Yes/No? y

(parted) print
Model: Kingston DataTraveler 2.0 (scsi)
Disk /dev/sdb: 1999MB
Sector size (logical/physical): 512B/512B
Partition Table: gpt
Disk Flags:

Number  Start   End     Size    File system  Name     Flags
 1      1049kB  99.6MB  98.6MB  ext4         primary
 2      99.6MB  200MB   100MB   ext4
 3      200MB   300MB   99.7MB  ext4         primary
```

TIP En lugar de especificar un nuevo tamaño, puede usar el parámetro `-M` de `resize2fs` para ajustar el tamaño del sistema de archivos de modo que sea lo suficientemente grande para los archivos que contiene.

Creación de particiones de intercambio

En Linux, el sistema puede intercambiar páginas de memoria de RAM a disco según sea necesario, almacenándolas en un espacio separado generalmente implementado como una partición separada en un disco, llamada *partición de intercambio* o simplemente *intercambio*. Esta partición debe ser de un tipo específico y configurarse con una utilidad adecuada (`mkswap`) antes de que pueda usarse.

Para crear la partición de intercambio usando `fdisk` o `gdisk`, simplemente proceda como si estuviera creando una partición normal, como se explicó antes. La única diferencia es que necesitará cambiar el tipo de partición a *Linux swap*.

- En `fdisk` use el comando `t`. Seleccione la partición que desea utilizar y cambie su tipo a `82`. Escriba los cambios en el disco y salga con `w`.

- En `gdisk` el comando para cambiar el tipo de partición también es `t`, pero el código es `8200`. Escriba los cambios en el disco y salga con `w`.

Si está usando `parted`, la partición debe identificarse como una partición de intercambio durante la creación, simplemente use `linux-swap` como tipo de sistema de archivos. Por ejemplo, el comando para crear una partición de intercambio de 500 MB a partir de 300 MB en el disco es:

```
(parted) mkpart primary linux-swap 301m 800m
```

Una vez que la partición está creada e identificada correctamente, simplemente use `mkswap` seguido del dispositivo que representa la partición que desea usar, como:

```
# mkswap /dev/sda2
```

Para habilitar el intercambio en esta partición, use `swapon` seguido del nombre del dispositivo:

```
# swapon /dev/sda2
```

Del mismo modo, `swapoff`, seguido del nombre del dispositivo, desactivará el intercambio en ese dispositivo.

Linux también admite el uso de swap *files* en lugar de particiones. Simplemente cree un archivo vacío del tamaño que desee usando `dd` y luego use `mkswap` y `swapon` con este archivo como destino.

Los siguientes comandos crearán un archivo de 1 GB llamado `myswap` en el directorio actual, lleno de ceros, y luego lo configurarán y habilitarán como un archivo de intercambio.

Cree el archivo de intercambio:

```
$ dd if=/dev/zero of=myswap bs=1M count=1024
1024+0 records in
1024+0 records out
1073741824 bytes (1.1 GB, 1.0 GiB) copied, 7.49254 s, 143 MB/s
```

`if=` es el archivo de entrada, el origen de los datos que se escribirán en el archivo. En este caso es el dispositivo `/dev/zero`, que proporciona tantos caracteres `NULL` como se soliciten. `of=` es el archivo de salida, el archivo que se creará. `bs=` es el tamaño de los bloques de datos, aquí especificado en Megabytes, y `count=` es la cantidad de bloques que se escribirán en la salida. 1024 bloques de 1 MB

cada uno equivale a 1 GB.

```
# mkswap myswap
Setting up swapspace version 1, size = 1024 MiB (1073737728 bytes)
no label, UUID=49c53bc4-c4b1-4a8b-a613-8f42cb275b2b

# swapon myswap
```

Usando los comandos anteriores, este archivo de intercambio se usará solo durante la sesión actual del sistema. Si se reinicia la máquina, el archivo seguirá estando disponible, pero no se cargará automáticamente. Puede automatizar eso agregando el nuevo archivo de intercambio a `/etc/fstab`, que discutiremos en una lección posterior.

TIP Tanto `mkswap` como `swapon` se quejarán si su archivo de intercambio tiene permisos inseguros. El indicador de permiso de archivo recomendado es `0600`. El propietario y el grupo deben ser `root`.

Ejercicios Guiados

1. ¿Qué esquema de partición debería utilizarse para particionar un disco duro de 3 TB en tres particiones de 1 GB? ¿Por qué?

2. En `gdisk`, ¿cómo podemos saber cuánto espacio hay disponible en el disco?

3. ¿Cuál sería el comando para crear un sistema de archivos ext3, verificando antes los bloques defectuosos, con la etiqueta `MyDisk` y un UUID aleatorio, en el dispositivo `/dev/sdc1`?

4. Usando `parted`, ¿cuál es el comando para crear una partición ext4 de 300 MB, comenzando con 500 MB en el disco?

5. Imagine que tiene 2 particiones, una en `/dev/sda1` y la otra en `/dev/sda2`, ambas de 20 GB de tamaño. ¿Cómo puede usarlas en un solo sistema de archivos Btrfs, de tal manera que el contenido de una partición se refleje automáticamente en la otra, como en una configuración RAID1? ¿Qué tamaño tendrá el sistema de archivos?

Ejercicios Exploratorios

1. Considere un disco de 2 GB con una tabla de particiones MBR y el siguiente diseño:

```
Disk /dev/sdb: 1.9 GiB, 1998631936 bytes, 3903578 sectors
Disk model: DataTraveler 2.0
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: dos
Disk identifier: 0x31a83a48

Device     Boot   Start     End Sectors  Size Id Type
/dev/sdb1          2048 1050623 1048576  512M 83 Linux
/dev/sdb3       2099200 3147775 1048576  512M 83 Linux
```

   ¿Puede crear una partición de 600 MB en él? ¿Por qué?

2. En un disco en `/dev/sdc`, tenemos una primera partición de 1 GB, que contiene aproximadamente 256 MB de archivos. Usando `parted`, ¿cómo puede reducirlo para que tenga suficiente espacio para los archivos?

3. Imagine que tiene un disco en `/dev/sdb` y desea crear una partición de intercambio de 1 GB al principio. Entonces, usando `parted`, cree la partición con `mkpart primary linux-swap 0 1024M`. Luego, habilita el intercambio en esta partición con `swapon /dev/sdb1`, pero aparece el siguiente mensaje de error:

   `swapon: /dev/sdb1: read swap header failed`

   ¿Qué salió mal?

4. Durante el curso de esta lección, estuvo probando algunos comandos en `parted` pero, por error, eliminó la tercera partición en su disco duro. Usted sabe que vino después de una partición UEFI de 250 MB y una partición de intercambio de 4 GB, y tenía un tamaño de 10 GB. ¿Qué comando puede usar para recuperarlo?

5. Imagine que tiene una partición de 4 GB sin usar en `/dev/sda3`. Usando `fdisk`, ¿cuál sería la secuencia de operaciones para convertirlo en una partición de intercambio activa?

Resumen

En esta lección aprendimos:

- Cómo crear una tabla de particiones MBR en un disco con `fdisk`, y cómo usarlo para crear y eliminar particiones
- Cómo crear una tabla de particiones MBR en un disco con `gdisk`, y cómo usarlo para crear y eliminar particiones
- Cómo crear particiones ext2, ext3, ext4, XFS, VFAT y exFAT
- Cómo utilizar `parted` para crear, eliminar y recuperar particiones en discos MBR y GPT
- Cómo usar las particiones resize ext2, ext3, ext4 y Brts
- Cómo crear, configurar y activar particiones de intercambio y archivos de intercambio

Los siguientes comandos se discutieron en esta lección:

- `fdisk`
- `gdisk`
- `mkfs.ext2`, `mkfs.ext3`, `mkfs.ext4`, `mkfs.xfs`, `mkfs.vfat` y `mkfs.exfat`
- `parted`
- `btrfs`
- `mkswap`
- `swapon` y `swapoff`

Respuestas a los ejercicios guiados

1. ¿Qué esquema de partición debería utilizarse para particionar un disco duro de 3 TB en tres particiones de 1 GB? ¿Por qué?

   GPT, ya que MBR admite como máximo discos duros de 2 TB.

2. En `gdisk`, ¿cómo podemos saber cuánto espacio hay disponible en el disco?

   Utilice `p` (print). El espacio libre total se mostrará como la última línea de información antes de la tabla de particiones.

3. ¿Cuál sería el comando para crear un sistema de archivos ext3, buscando bloques defectuosos antes, con la etiqueta `MyDisk` y un UUID aleatorio, en el dispositivo `/dev/sdc1`?

   El comando sería `mkfs.ext3 -c -L MyDisk -U random /dev/sdc1`. Alternativamente, `mke2fs -t ext3` también se puede usar en lugar de `mkfs.ext3`

4. Usando `parted`, ¿cuál es el comando para crear una partición ext4 de 300 MB, comenzando con 500 MB en el disco?

   Utilice `mkpart primary ext4 500m 800m`. Recuerde que tendrá que crear el sistema de archivos usando `mkfs.ext4`, ya que parted no hace esto.

5. Imagine que tiene 2 particiones, una en `/dev/sda1` y la otra en `/dev/sda2`, ambas de 20 GB de tamaño. ¿Cómo puede usarlas en un solo sistema de archivos Btrfs, de tal manera que el contenido de una partición se refleje automáticamente en la otra, como en una configuración RAID1? ¿Qué tamaño tendrá el sistema de archivos?

   Utilice `mkfs.btrfs /dev/sda1 /dev/sdb1 -m raid1`. El sistema de archivos resultante tendrá un tamaño de 20 GB, ya que una partición actúa simplemente como un espejo de la otra.

Respuestas a ejercicios exploratorios

1. Considere un disco de 2 GB con una tabla de particiones MBR y el siguiente diseño:

```
Disk /dev/sdb: 1.9 GiB, 1998631936 bytes, 3903578 sectors
Disk model: DataTraveler 2.0
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: dos
Disk identifier: 0x31a83a48

Device     Boot   Start     End Sectors  Size Id Type
/dev/sdb1          2048 1050623 1048576  512M 83 Linux
/dev/sdb3       2099200 3147775 1048576  512M 83 Linux
```

¿Puede crear una partición de 600 MB en él? ¿Por qué?

No puede, porque no hay suficiente espacio contiguo. La primera pista de que algo está “no coincide” es la lista de dispositivos: tiene `/dev/sdb1` y `/dev/sdb3`, pero no `/dev/sdb2`. Entonces, falta algo.

Luego, debe mirar dónde termina una partición y comienza la otra. La partición uno termina en el sector `1050623` y la partición 2 comienza en `2099200`. Esa es una “brecha” de 1048577 sectores. A 512 bytes por sector, eso es 536.871.424 bytes. Si lo divide por 1024 obtiene 524.288 Kilobytes. Divida por 1024 de nuevo y obtendrá … 512 MB. Este es el tamaño de la “brecha”.

Si el disco tiene 2 GB, entonces moveríamos como máximo otros 512 MB después de la partición 3. Incluso si tenemos en total alrededor de 1 GB sin asignar, el bloque contiguo más grande es de 512 MB. Entonces, no hay espacio para una partición de 600 MB.

2. En un disco en `/dev/sdc`, tenemos una primera partición de 1 GB, que contiene aproximadamente 256 MB de archivos. Usando `parted`, ¿cómo puede reducirlo para que tenga suficiente espacio para los archivos?

Esta es una operación de varias partes. Primero tiene que encoger el sistema de archivos usando `resize2fs`. En lugar de especificar el nuevo tamaño directamente, puede utilizar el parámetro `-M` para que sea “suficientemente grande”. Entonces: `resize2fs -M /dev/sdc1`.

Luego, cambie el tamaño de la partición con parted usando `resizepart`. Dado que es la primera partición, podemos asumir que comienza en cero y termina en 241 MB. Entonces, el comando es `resizepart 1 241M`.

3. Imagine que tiene un disco en `/dev/sdb` y desea crear una partición de intercambio de 1 GB al principio. Entonces, usando `parted`, crea la partición con `mkpart primary linux-swap 0 1024M`. Luego, habilita el intercambio en esta partición con `swapon /dev/sdb1`, pero aparece el siguiente mensaje de error:

   ```
   swapon: /dev/sdb1: read swap header failed
   ```

   ¿Qué salió mal?

   Creó una partición del tipo correcto (`linux-swap`), pero recuerde que `mkpart` *no crea un sistema de archivos*. Olvidó configurar la partición como un espacio de intercambio primero con `mkswap` antes de usarla.

4. Durante el curso de esta lección, estuvo probando algunos comandos en `parted` pero, por error, eliminó la tercera partición en su disco duro. Usted sabe que vino después de una partición UEFI de 250 MB y una partición de intercambio de 4 GB, y tenía un tamaño de 10 GB. ¿Qué comando puede usar para recuperarlo?

   Tranquilo, tiene toda la información que necesita para recuperar la partición, solo use `rescue` y haga los cálculos. Tenía 250 MB + 4.096 MB (4*1024) antes, por lo que el punto de inicio debería ser alrededor de 4346 MB. Más 10.240 MB (10*1024) de tamaño, debería terminar en 14.586 MB. Entonces, `rescue 4346m 14586m` debería ser el truco. Es posible que deba dar un poco de “espacio adicional” para rescatar, comenzando un poco antes y terminando un poco tarde, dependiendo de la geometría de su disco.

5. Imagine que tiene una partición de 4 GB sin usar en `/dev/sda3`. Usando `fdisk`, ¿cuál sería la secuencia de operaciones para convertirlo en una partición de intercambio activa?

   Primero, cambie el tipo de partición a “Linux Swap”(82), escriba sus cambios en el disco y salga. Luego, use `mkswap` para configurar la partición como un área de intercambio. Luego, use `swapon` para habilitarlo.



104.2 Mantener la integridad de los sistemas de archivos

Reference to LPI objectives

LPIC-1 version 5.0, Exam 101, Objective 104.2

Weight

2

Key knowledge areas

- Verificar la integridad de los sistemas de archivos.
- Supervisar el espacio libre y los inodos.
- Solucionar problemas simples relacionados con los sistemas de archivos.

Partial list of the used files, terms and utilities

- `du`
- `df`
- `fsck`
- `e2fsck`
- `mke2fs`
- `tune2fs`
- `xfs_repair`
- `xfs_fsr`
- `xfs_db`



104.2 Lección 1

Certificación:	LPIC-1
Versión:	5.0
Tema:	104 Dispositivos, sistemas de archivos Linux, estándar de jerarquía del sistema de archivos
Objetivo:	104.2 Mantener la integridad de los sistemas de archivos.
Lección:	1 de 1

Introducción

Los sistemas de archivos modernos de Linux utilizan *journaling*. Esto significa que cada operación se refleja en un registro interno (el *journal*) antes de ejecutarse. Si la operación se interrumpe debido a un error del sistema (como un pánico del kernel, falla de energía, etc.), se puede reconstruir revisando el diario, evitando la corrupción del sistema de archivos y la pérdida de datos.

Esto reduce en gran medida la necesidad de verificaciones manuales del sistema de archivos, pero es posible que aún sean necesarias. Conocer las herramientas utilizadas para esto (y los parámetros correspondientes) puede representar la diferencia entre cenar en casa con tu familia o pasar la noche en la sala de servidores del trabajo.

En esta lección, discutiremos las herramientas disponibles para monitorear el uso del sistema de archivos, optimizar su operación y cómo verificar y reparar daños.

Comprobación del uso del disco

Hay dos comandos que se pueden usar para verificar cuánto espacio se está usando y cuánto queda en un sistema de archivos. El primero es `du`, que significa "uso de disco".

`du` es de naturaleza recursiva. En su forma más básica, el comando simplemente mostrará cuántos bloques de 1 Kilobyte están siendo utilizados por el directorio actual y todos sus subdirectorios:

```
$ du
4816    .
```

Esto no es muy útil, por lo que podemos solicitar más salida "legible por humanos" agregando el parámetro `-h`:

```
$ du -h
4.8M    .
```

Por defecto, `du` solo muestra el recuento de uso de los directorios (considerando todos los archivos y subdirectorios que contiene). Para mostrar un recuento individual de todos los archivos en el directorio, use el parámetro `-a`:

```
$ du -ah
432K    ./geminoid.jpg
508K    ./Linear_B_Hero.jpg
468K    ./LG-G8S-ThinQ-Mirror-White.jpg
656K    ./LG-G8S-ThinQ-Range.jpg
60K ./Stranger3_Titulo.png
108K    ./Baidu_Banho.jpg
324K    ./Xiaomi_Mimoji.png
284K    ./Mi_CC_9e.jpg
96K ./Mimoji_Comparativo.jpg
32K ./Xiaomi FCC.jpg
484K    ./geminoid2.jpg
108K    ./Mimoji_Abre.jpg
88K ./Mi8_Hero.jpg
832K    ./Tablet_Linear_B.jpg
332K    ./Mimoji_Comparativo.png
4.8M    .
```

El comportamiento predeterminado es mostrar el uso de cada subdirectorio, luego el uso total del

directorio actual, *incluyendo* subdirectorios:

```
$ du -h
4.8M    ./Temp
6.0M    .
```

En el ejemplo anterior, podemos ver que el subdirectorio `Temp` ocupa 4.8 MB y el directorio actual, *incluyendo* `Temp`, ocupa 6.0 MB. Pero, ¿cuánto espacio ocupan los *archivos* en el directorio actual, excluyendo los subdirectorios? Para eso tenemos el parámetro `-S`:

```
$ du -Sh
4.8M    ./Temp
1.3M    .
```

TIP Tenga en cuenta que los parámetros de la línea de comandos distinguen entre mayúsculas y minúsculas: `-s` es diferente de `-S`.

Si desea mantener esta distinción entre el espacio usado por los archivos en el directorio actual y el espacio usado por los subdirectorios, pero *también* quiere un gran total al final, puede agregar el parámetro `-c`:

```
$ du -Shc
4.8M    ./Temp
1.3M    .
6.0M    total
```

Puede controlar qué tan "profundo" debe ir la salida de `du` con el parámetro `-d N`, donde `N` describe los niveles. Por ejemplo, si usa el parámetro `-d 1`, mostrará el directorio actual y sus subdirectorios, pero no los subdirectorios de esos.

Vea la diferencia a continuación. Sin `-d`:

```
$ du -h
216K    ./somedir/anotherdir
224K    ./somedir
232K    .
```

Y limitando la profundidad a un nivel con `-d 1`:

```
$ du -h -d1
224K    ./somedir
232K    .
```

Tenga en cuenta que incluso si no se muestra `anotherdir`, su tamaño se sigue teniendo en cuenta.

Es posible que desee excluir algunos tipos de archivos del recuento, lo que puede hacer con `--exclude="PATTERN"`, donde `PATTERN` es el patrón con el que desea comparar. Considere este directorio:

```
$ du -ah
124K    ./ASM68K.EXE
2.0M    ./Contra.bin
36K ./fixheadr.exe
4.0K    ./README.txt
2.1M    ./Contra_NEW.bin
4.0K    ./Built.bat
8.0K    ./Contra_Main.asm
4.2M    .
```

Ahora, usaremos `--exclude` para filtrar todos los archivos con la extensión `.bin`:

```
$ du -ah --exclude="*.bin"
124K    ./ASM68K.EXE
36K ./fixheadr.exe
4.0K    ./README.txt
4.0K    ./Built.bat
8.0K    ./Contra_Main.asm
180K    .
```

Tenga en cuenta que el total ya no refleja el tamaño de los archivos excluidos.

Comprobación de espacio libre

`du` funciona a nivel de archivos. Hay otro comando que puede mostrarle el uso del disco y la cantidad de espacio disponible a nivel del sistema de archivos. Este comando es `df`.

El comando `df` proporcionará una lista de todos los sistemas de archivos disponibles (ya montados) en su sistema, incluido su tamaño total, cuánto espacio se ha utilizado, cuánto espacio está disponible, el porcentaje de uso y dónde está montado:

```
$ df
Filesystem     1K-blocks      Used Available Use% Mounted on
udev             2943068         0   2943068   0% /dev
tmpfs             595892      2496    593396   1% /run
/dev/sda1      110722904  25600600  79454800  25% /
tmpfs            2979440    951208   2028232  32% /dev/shm
tmpfs               5120         0      5120   0% /run/lock
tmpfs            2979440         0   2979440   0% /sys/fs/cgroup
tmpfs             595888        24    595864   1% /run/user/119
tmpfs             595888       116    595772   1% /run/user/1000
/dev/sdb1          89111      1550     80824   2% /media/carol/part1
/dev/sdb3          83187      1550     75330   3% /media/carol/part3
/dev/sdb2          90827      1921     82045   3% /media/carol/part2
/dev/sdc1      312570036 233740356  78829680  75% /media/carol/Samsung Externo
```

Sin embargo, mostrar el tamaño en bloques de 1 KB no es muy fácil de usar. Como en `du`, puede agregar los parámetros `-h` para obtener un resultado más “legible por humanos”:

```
$ df -h
Filesystem      Size  Used Avail Use% Mounted on
udev            2.9G     0  2.9G   0% /dev
tmpfs           582M  2.5M  580M   1% /run
/dev/sda1       106G   25G   76G  25% /
tmpfs           2.9G  930M  2.0G  32% /dev/shm
tmpfs           5.0M     0  5.0M   0% /run/lock
tmpfs           2.9G     0  2.9G   0% /sys/fs/cgroup
tmpfs           582M   24K  582M   1% /run/user/119
tmpfs           582M  116K  582M   1% /run/user/1000
/dev/sdb1        88M  1.6M   79M   2% /media/carol/part1
/dev/sdb3        82M  1.6M   74M   3% /media/carol/part3
/dev/sdb2        89M  1.9M   81M   3% /media/carol/part2
/dev/sdc1       299G  223G   76G  75% /media/carol/Samsung Externo
```

También puede usar el parámetro `-i` para mostrar inodos usados/disponibles en lugar de bloques:

```
$ df -i
Filesystem      Inodes  IUsed   IFree IUse% Mounted on
udev            737142    547  736595    1% /dev
tmpfs           745218    908  744310    1% /run
/dev/sda6      6766592 307153 6459439    5% /
tmpfs           745218    215  745003    1% /dev/shm
tmpfs           745218      4  745214    1% /run/lock
tmpfs           745218     18  745200    1% /sys/fs/cgroup
/dev/sda1        62464    355   62109    1% /boot
tmpfs           745218     43  745175    1% /run/user/1000
```

Un parámetro útil es `-T`, que también imprimirá el tipo de cada sistema de archivos:

```
$ df -hT
Filesystem     Type      Size  Used Avail Use% Mounted on
udev           devtmpfs  2.9G     0  2.9G   0% /dev
tmpfs          tmpfs     582M  2.5M  580M   1% /run
/dev/sda1      ext4      106G   25G   76G  25% /
tmpfs          tmpfs     2.9G  930M  2.0G  32% /dev/shm
tmpfs          tmpfs     5.0M     0  5.0M   0% /run/lock
tmpfs          tmpfs     2.9G     0  2.9G   0% /sys/fs/cgroup
tmpfs          tmpfs     582M   24K  582M   1% /run/user/119
tmpfs          tmpfs     582M  116K  582M   1% /run/user/1000
/dev/sdb1      ext4       88M  1.6M   79M   2% /media/carol/part1
/dev/sdb3      ext4       82M  1.6M   74M   3% /media/carol/part3
/dev/sdb2      ext4       89M  1.9M   81M   3% /media/carol/part2
/dev/sdc1      fuseblk   299G  223G   76G  75% /media/carol/Samsung Externo
```

Conociendo el tipo de sistema de archivos, puede filtrar la salida. Puede mostrar solo sistemas de archivos de un tipo dado con `-t TYPE`, o excluir sistemas de archivos de un tipo dado con `-x TYPE`, como en los ejemplos siguientes.

Excluyendo los sistemas de archivos `tmpfs`:


```
$ df -hx tmpfs
Filesystem      Size  Used Avail Use% Mounted on
udev            2.9G     0  2.9G   0% /dev
/dev/sda1       106G   25G   76G  25% /
/dev/sdb1        88M  1.6M   79M   2% /media/carol/part1
/dev/sdb3        82M  1.6M   74M   3% /media/carol/part3
/dev/sdb2        89M  1.9M   81M   3% /media/carol/part2
/dev/sdc1       299G  223G   76G  75% /media/carol/Samsung Externo
```

Mostrando solo sistemas de archivos `ext4`:

```
$ df -ht ext4
Filesystem      Size  Used Avail Use% Mounted on
/dev/sda1       106G   25G   76G  25% /
/dev/sdb1        88M  1.6M   79M   2% /media/carol/part1
/dev/sdb3        82M  1.6M   74M   3% /media/carol/part3
/dev/sdb2        89M  1.9M   81M   3% /media/carol/part2
```

También puede personalizar la salida de `df`, seleccionando lo que debe mostrarse y en qué orden, usando el parámetro `--output=` seguido de una lista de campos separados por comas que desee mostrar. Algunos de los campos disponibles son:

`source`

El dispositivo correspondiente al sistema de archivos.

`fstype`

El tipo de sistema de archivos.

`size`

El tamaño total del sistema de archivos.

`used`

Cuánto espacio se está utilizando.

`avail`

Cuánto espacio hay disponible.

`pcent`

El porcentaje de uso.

`target`
: Dónde está montado el sistema de archivos (punto de montaje).

Si desea una salida que muestre el destino, la fuente, el tipo y el uso, puede usar:

```
$ df -h --output=target,source,fstype,pcent
Mounted on                    Filesystem     Type     Use%
/dev                          udev           devtmpfs   0%
/run                          tmpfs          tmpfs      1%
/                             /dev/sda1      ext4      25%
/dev/shm                      tmpfs          tmpfs     32%
/run/lock                     tmpfs          tmpfs      0%
/sys/fs/cgroup                tmpfs          tmpfs      0%
/run/user/119                 tmpfs          tmpfs      1%
/run/user/1000                tmpfs          tmpfs      1%
/media/carol/part1            /dev/sdb1      ext4       2%
/media/carol/part3            /dev/sdb3      ext4       3%
/media/carol/part2            /dev/sdb2      ext4       3%
/media/carol/Samsung Externo  /dev/sdc1      fuseblk   75%
```

`df` también se puede usar para verificar la información de inodos, pasando los siguientes campos a `--output=`:

`itotal`
: El número total de inodos en el sistema de archivos.

`iused`
: El número de inodos usados en el sistema de archivos.

`iavail`
: El número de inodos disponibles en el sistema de archivos.

`ipcent`
: El porcentaje de inodos usados en el sistema de archivos.

Por ejemplo:

```
$ df --output=source,fstype,itotal,iused,ipcent
Filesystem     Type       Inodes  IUsed IUse%
udev           devtmpfs  735764    593     1%
tmpfs          tmpfs     744858   1048     1%
/dev/sda1      ext4     7069696 318651     5%
tmpfs          tmpfs     744858    222     1%
tmpfs          tmpfs     744858      3     1%
tmpfs          tmpfs     744858     18     1%
tmpfs          tmpfs     744858     22     1%
tmpfs          tmpfs     744858     40     1%
```

Mantenimiento de los sistemas de archivos ext2, ext3 y ext4

Para comprobar un sistema de archivos en busca de errores (y con suerte corregirlos), Linux proporciona la utilidad `fsck` (piense en "filesystem check" y nunca olvidará el nombre). En su forma más básica, puede invocarlo con `fsck` seguido de la ubicación del sistema de archivos que desee verificar:

```
# fsck /dev/sdb1
fsck from util-linux 2.33.1
e2fsck 1.44.6 (5-Mar-2019)
DT_2GB: clean, 20/121920 files, 369880/487680 blocks
```

WARNING NUNCA ejecute `fsck` (o utilidades relacionadas) en un sistema de archivos montado. Si lo hace de todos modos, es posible que se pierdan los datos.

`fsck` por sí mismo no verificará el sistema de archivos, simplemente llamará a la utilidad apropiada para el tipo de sistema de archivos para hacerlo. En el ejemplo anterior, dado que no se especificó un tipo de sistema de archivos, `fsck` asumió un sistema de archivos ext2/3/4 por defecto y se llamó `e2fsck`.

Para especificar un sistema de archivos, use la opción `-t`, seguida del nombre del sistema de archivos, como en `fsck -t vfat /dev/sdc`. Alternativamente, puede llamar directamente a una utilidad específica del sistema de archivos, como `fsck.msdos` para sistemas de archivos FAT.

TIP Escriba `fsck` seguido de *Tab* dos veces para ver una lista de todos los comandos relacionados en su sistema.

`fsck` puede tomar algunos argumentos de línea de comandos. Estos son algunos de los más comunes:

`-A`
: Esto comprobará todos los sistemas de archivos listados en `/etc/fstab`.

`-C`
: Muestra una barra de progreso al comprobar un sistema de archivos. Actualmente solo funciona en sistemas de archivos ext2/3/4.

`-N`
: Esto imprimirá lo que se haría y saldrá, sin realmente verificar el sistema de archivos.

`-R`
: Cuando se usa junto con `-A`, esto omitirá la verificación del sistema de archivos raíz.

`-V`
: Modo detallado, imprime más información de lo habitual durante el funcionamiento. Esto es útil para depurar.

La utilidad específica para los sistemas de archivos ext2, ext3 y ext4 es `e2fsck`, también llamado `fsck.ext2`, `fsck.ext3` y `fsck.ext4` (esos tres son simplemente enlaces a `e2fsck`). De forma predeterminada, se ejecuta en modo interactivo: cuando se encuentra un error en el sistema de archivos, se detiene y le pregunta al usuario qué hacer. El usuario debe escribir `y` para solucionar el problema, `n` para dejarlo sin arreglar o `a` para solucionar el problema actual y todos los posteriores.

Por supuesto, sentarse frente a una terminal esperando que `e2fsck` pregunte qué hacer no es un uso productivo de su tiempo, especialmente si se trata de un sistema de archivos grande. Entonces, hay opciones que hacen que `e2fsck` se ejecute en modo no interactivo:

`-p`
: Esto intentará corregir automáticamente cualquier error encontrado. Si se encuentra un error que requiere la intervención del administrador del sistema, `e2fsck` proporcionará una descripción del problema y saldrá.

`-y`
: Esto responderá `y` (sí) a todas las preguntas.

`-n`
: Lo contrario de `-y`. Además de responder `n` (no) a todas las preguntas, esto hará que el sistema de archivos se monte como de solo lectura, por lo que no se puede modificar.

-f

Obliga a `e2fsck` a comprobar un sistema de archivos incluso si está marcado como "clean", es decir, se ha desmontado correctamente.

Ajustes de un sistema de archivos ext

Los sistemas de archivos ext2, ext3 y ext4 tienen una serie de parámetros que el administrador del sistema puede ajustar o "afinar" para adaptarse mejor a las necesidades del sistema. La utilidad utilizada para mostrar o modificar estos parámetros se llama `tune2fs`.

Para ver los parámetros actuales para cualquier sistema de archivos dado, use el parámetro `-l` seguido del dispositivo que representa la partición. El siguiente ejemplo muestra el resultado de este comando en la primera partición del primer disco (`/dev/sda1`) de una máquina:

```
# tune2fs -l /dev/sda1
tune2fs 1.44.6 (5-Mar-2019)
Filesystem volume name:   <none>
Last mounted on:          /
Filesystem UUID:          6e2c12e3-472d-4bac-a257-c49ac07f3761
Filesystem magic number:  0xEF53
Filesystem revision #:    1 (dynamic)
Filesystem features:      has_journal ext_attr resize_inode dir_index filetype
needs_recovery extent 64bit flex_bg sparse_super large_file huge_file dir_nlink
extra_isize metadata_csum
Filesystem flags:         signed_directory_hash
Default mount options:    user_xattr acl
Filesystem state:         clean
Errors behavior:          Continue
Filesystem OS type:       Linux
Inode count:              7069696
Block count:              28255605
Reserved block count:     1412780
Free blocks:              23007462
Free inodes:              6801648
First block:              0
Block size:               4096
Fragment size:            4096
Group descriptor size:    64
Reserved GDT blocks:      1024
Blocks per group:         32768
Fragments per group:      32768
Inodes per group:         8192
Inode blocks per group:   512
```

```
Flex block group size:    16
Filesystem created:       Mon Jun 17 13:49:59 2019
Last mount time:          Fri Jun 28 21:14:38 2019
Last write time:          Mon Jun 17 13:53:39 2019
Mount count:              8
Maximum mount count:      -1
Last checked:             Mon Jun 17 13:49:59 2019
Check interval:           0 (<none>)
Lifetime writes:          20 GB
Reserved blocks uid:      0 (user root)
Reserved blocks gid:      0 (group root)
First inode:              11
Inode size:           256
Required extra isize:     32
Desired extra isize:      32
Journal inode:            8
First orphan inode:       5117383
Default directory hash:   half_md4
Directory Hash Seed:      fa95a22a-a119-4667-a73e-78f77af6172f
Journal backup:           inode blocks
Checksum type:            crc32c
Checksum:                 0xe084fe23
```

Los sistemas de archivos ext tienen *conteos de montajes*. El recuento aumenta en 1 cada vez que se monta el sistema de archivos, y cuando se alcanza un valor de umbral (el *recuento máximo de montaje*), el sistema se comprobará automáticamente con `e2fsck` en el próximo arranque.

El número máximo de montajes se puede establecer con el parámetro `-c N`, donde `N` es el número de veces que se puede montar el sistema de archivos sin comprobarlo. El parámetro `-C N` establece el número de veces que se ha montado el sistema en el valor de `N`. Tenga en cuenta que los parámetros de la línea de comandos distinguen entre mayúsculas y minúsculas, por lo que `-c` es diferente de `-C`.

También es posible definir un intervalo de tiempo entre comprobaciones, con el parámetro `-i`, seguido de un número y las letras `d` para días, `m` para meses e `y` para años. Por ejemplo, `-i 10d` comprobaría el sistema de archivos en el próximo reinicio cada 10 días. Utilice cero como valor para desactivar esta función.

`-L` se puede usar para establecer una etiqueta para el sistema de archivos. Esta etiqueta puede tener hasta 16 caracteres. El parámetro `-U` establece el UUID para el sistema de archivos, que es un número hexadecimal de 128 bits. En el ejemplo anterior, el UUID es `6e2c12e3-472d-4bac-a257-c49ac07f3761`. Tanto la etiqueta como el UUID pueden usarse en lugar del nombre del dispositivo (como `/dev/sda1`) para montar el sistema de archivos.

La opción `-e BEHAVIOUR` define el comportamiento del kernel cuando se encuentra un error en el sistema de archivos. Hay tres posibles comportamientos:

`continue`
: Continuará la ejecución normalmente.

`remount-ro`
: Volverá a montar el sistema de archivos como de solo lectura.

`panic`
: Causará *kernel panic*.

El comportamiento predeterminado es `continuar`. `remount-ro` podría ser útil en aplicaciones sensibles a los datos, ya que detendrá inmediatamente las escrituras en el disco, evitando más errores potenciales.

Los sistemas de archivos ext3 son básicamente sistemas de archivos ext2 con un journal. Usando `tune2fs` puede agregar un journal a un sistema de archivos ext2, convirtiéndolo así en ext3. El procedimiento es simple, simplemente pase el parámetro `-j` a `tune2fs`, seguido del dispositivo que contiene el sistema de archivos:

```
# tune2fs -j /dev/sda1
```

Luego, cuando monte el sistema de archivos convertido, no olvide establecer el tipo en `ext3` para que se pueda usar el journal.

Cuando se trata de sistemas de archivos registrados por journal, el parámetro `-J` le permite usar parámetros adicionales para establecer algunas opciones de journal, como `-J size=` para establecer el tamaño del journal (en megabytes), `-J location=` para especificar dónde el journal debe almacenarse (ya sea en un bloque específico o en una posición específica en el disco con sufijos como `M` o `G`) e incluso poner el journal en un dispositivo externo con `-J device=`.

Puede especificar varios parámetros a la vez separándolos con una coma. Por ejemplo: `-J size=10,location=100M,device=/dev/sdb1` creará un journal de 10 MB en la posición de 100 MB en el dispositivo `/dev/sdb1`.

WARNING

`tune2fs` tiene una opción de “fuerza bruta”, `-f`, que lo obligará a completar una operación incluso si se encuentran errores. No hace falta decir que esto solo debe usarse con extrema precaución.

Mantenimiento de sistemas de archivos XFS

Para los sistemas de archivos XFS, el equivalente de `fsck` es `xfs_repair`. Si sospecha que algo anda mal con el sistema de archivos, lo primero que debe hacer es escanearlo en busca de daños.

Esto se puede hacer pasando el parámetro `-n` a `xfs_repair`, seguido del dispositivo que contiene el sistema de archivos. El parámetro `-n` significa "no modificar": se comprobará el sistema de archivos, se informarán los errores pero no se realizarán reparaciones:

```
# xfs_repair -n /dev/sdb1
Phase 1 - find and verify superblock...
Phase 2 - using internal log
        - zero log...
        - scan filesystem freespace and inode maps...
        - found root inode chunk
Phase 3 - for each AG...
        - scan (but do not clear) agi unlinked lists...
        - process known inodes and perform inode discovery...
        - agno = 0
        - agno = 1
        - agno = 2
        - agno = 3
        - process newly discovered inodes...
Phase 4 - check for duplicate blocks...
        - setting up duplicate extent list...
        - check for inodes claiming duplicate blocks...
        - agno = 1
        - agno = 3
        - agno = 0
        - agno = 2
No modify flag set, skipping phase 5
Phase 6 - check inode connectivity...
        - traversing filesystem ...
        - traversal finished ...
        - moving disconnected inodes to lost+found ...
Phase 7 - verify link counts...
No modify flag set, skipping filesystem flush and exiting.
```

Si se encuentran errores, puede proceder a hacer las reparaciones sin el parámetro `-n`, así: `xfs_repair /dev/sdb1`.

`xfs_repair` acepta varias opciones de línea de comandos. Entre ellos:

`-l LOGDEV` y `-r RTDEV`

Estos son necesarios si el sistema de archivos tiene un registro externo y secciones en tiempo real. En este caso, reemplace `LOGDEV` y `RTDEV` con los dispositivos correspondientes.

`-m N`

Se utiliza para limitar el uso de memoria de `xfs_repair` a `N` megabytes, algo que puede ser útil en la configuración del servidor. Según la página del manual, de forma predeterminada, `xfs_repair` escalará su uso de memoria según sea necesario, hasta el 75% de la RAM física del sistema.

`-d`

El modo “dangerous” permitirá la reparación de sistemas de archivos que estén montados como de solo lectura.

`-v`

Puede que lo haya adivinado: modo detallado. Cada vez que se usa este parámetro, la “verbosidad” aumenta (por ejemplo,`-v -v` imprimirá más información que solo `-v`).

Tenga en cuenta que `xfs_repair` no puede reparar sistemas de archivos con un registro “sucio”. Puede “poner a cero” un registro corrupto con el parámetro `-L`, pero tenga en cuenta que este es un *último recurso* ya que puede provocar la corrupción del sistema de archivos y la pérdida de datos.

Para depurar un sistema de archivos XFS, se puede usar la utilidad `xfs_db`, como en `xfs_db /dev/sdb1`. Esto se usa principalmente para inspeccionar varios elementos y parámetros del sistema de archivos.

Esta utilidad tiene un indicador interactivo, como `parted`, con muchos comandos internos. También hay disponible un sistema de ayuda: teclee `help` para ver una lista de todos los comandos, y `help` seguido del nombre del comando para ver más información sobre el comando.

Otra utilidad interesante es `xfs_fsr`, que se puede utilizar para reorganizar (“desfragmentar”) un sistema de archivos XFS. Cuando se ejecuta sin ningún argumento adicional, se ejecutará durante dos horas e intentará desfragmentar todos los sistemas de archivos XFS montados y escribibles enumerados en el archivo `/etc/mtab/`. Es posible que deba instalar esta utilidad utilizando el administrador de paquetes para su distribución de Linux, ya que es probable que no sea parte de una instalación predeterminada. Para más información consulte la página de manual correspondiente.

Ejercicios Guiados

1. Usando `du`, ¿cómo podemos verificar cuánto espacio están usando solo los archivos en el directorio actual?

2. Usando `df`, enumere la información para cada sistema de archivos ext4, con las salidas incluyendo los siguientes campos, en orden: dispositivo, punto de montaje, número total de inodos, número de inodos disponibles, porcentaje de espacio libre.

3. ¿Cuál es el comando para ejecutar `e2fsck` en `/dev/sdc1` en modo no interactivo, mientras se intenta corregir automáticamente la mayoría de los errores?

4. Suponga que `/dev/sdb1` es un sistema de archivos ext2. ¿Cómo puede convertirlo a ext3 y, al mismo tiempo, restablecer su recuento de montaje y cambiar su etiqueta a `UserData`?

5. ¿Cómo puede comprobar si hay errores en un sistema de archivos XFS, *sin* reparar los daños encontrados?

Ejercicios Exploratorios

1. Considere que tiene un sistema de archivos ext4 en `/dev/sda1` con los siguientes parámetros, obtenidos con `tune2fs`:

   ```
   Mount count:              8
   Maximum mount count:      -1
   ```

   ¿Qué pasará en el próximo arranque si se emite el comando `tune2fs -c 9 /dev/sda1`?

2. Considere la siguiente salida de `du -h`:

   ```
   $ du -h
   216K    ./somedir/anotherdir
   224K    ./somedir
   232K    .
   ```

   ¿Cuánto espacio ocupan solo los archivos en el directorio actual? ¿Cómo podríamos reescribir el comando para mostrar esta información con mayor claridad?

3. ¿Qué pasaría con el sistema de archivos ext2 `/dev/sdb1` si se emite el siguiente comando?

   ```
   # tune2fs -j /dev/sdb1 -J device=/dev/sdc1 -i 30d
   ```

4. ¿Cómo podemos verificar si hay errores en un sistema de archivos XFS en `/dev/sda1` que tiene una sección de registro en `/dev/sdc1`, sin realmente hacer ninguna reparación?

5. ¿Cuál es la diferencia entre los parámetros `-T` y `-t` para `df`?

Resumen

En esta lección aprendimos:

- Cómo comprobar si hay espacio libre y utilizado en un sistema de archivos.
- Cómo adaptar la salida de `df` a sus necesidades.
- Cómo verificar la integridad y reparar un sistema de archivos con `fsck` y `e2fsck`.
- Cómo ajustar un sistema de archivos ext con `tune2fs`.
- Cómo verificar y reparar sistemas de archivos XFS con `xfs_repair`.

Los siguientes comandos se discutieron en esta lección:

`du`
: Ver la cantidad de espacio en disco en uso en un sistema de archivos.

`df`
: Ver la cantidad de espacio en disco que está disponible (libre) en un sistema de archivos.

`fsck`
: La utilidad de reparación de verificación del sistema de archivos.

`e2fsck`
: La utilidad de reparación de verificación del sistema de archivos específica para sistemas de archivos extendidos (ext2/3/4).

`tune2fs`
: Modifica los parámetros del sistema de archivos en un sistema de archivos extendido (ext2/3/4).

`xfs_repair`
: El equivalente de `fsck` para sistemas de archivos XFS.

`xfs_db`
: Esta utilidad se utiliza para ver varios parámetros de un sistema de archivos XFS.

Respuestas a los ejercicios guiados

1. Usando `du`, ¿cómo podemos verificar cuánto espacio están usando solo los archivos en el directorio actual?

   Primero, use el parámetro `-S` para separar la salida del directorio actual de sus subdirectorios. Luego, use `-d 0` para limitar la profundidad de salida a cero, lo que significa “sin subdirectorios”. No olvide `-h` para obtener una salida en un formato “legible por humanos”:

   ```
   $ du -S -h -d 0
   ```

   o

   ```
   $ du -Shd 0
   ```

2. Usando `df`, enumere la información para cada sistema de archivos ext4, con las salidas incluyendo los siguientes campos, en orden: dispositivo, punto de montaje, número total de inodos, número de inodos disponibles, porcentaje de espacio libre.

   Puede filtrar sistemas de archivos con la opción `-t` seguida del nombre del sistema de archivos. Para obtener la salida necesaria, use `--output=source,target,itotal,iavail,pcent`. Entonces, la respuesta es:

   ```
   $ df -t ext4 --output=source,target,itotal,iavail,pcent
   ```

3. ¿Cuál es el comando para ejecutar `e2fsck` en `/dev/sdc1` en modo no interactivo, mientras se intenta corregir automáticamente la mayoría de los errores?

   El parámetro para intentar corregir automáticamente la mayoría de los errores es `-p`. Entonces la respuesta es:

   ```
   # e2fsck -p /dev/sdc1
   ```

4. Suponga que `/dev/sdb1` es un sistema de archivos ext2. ¿Cómo puede convertirlo a ext3 y al mismo tiempo restablecer su recuento de montaje y cambiar su etiqueta a `UserData`?

   Recuerde que convertir un sistema de archivos ext2 a ext3 es solo una cuestión de agregar un diario, lo cual se puede hacer con el parámetro `-j`. Para restablecer el recuento de monturas, use

`-C 0`. Para cambiar la etiqueta use `-L UserData`. La respuesta correcta es:

```
# tune2fs -j -C 0 -L UserData /dev/sdb1
```

5. ¿Cómo puede comprobar si hay errores en un sistema de archivos XFS, *sin* reparar los daños encontrados?

   Utilice el parámetro `-n`, como en `xfs -n`, seguido del dispositivo correspondiente.

Respuestas a ejercicios exploratorios

1. Considere que tiene un sistema de archivos ext4 en `/dev/sda1` con los siguientes parámetros, obtenidos con `tune2fs`:

   ```
   Mount count:              8
   Maximum mount count:      -1
   ```

   ¿Qué pasará en el próximo arranque si se emite el comando `tune2fs -c 9 /dev/sda1`?

   El comando establecerá el número máximo de montajes para el sistema de archivos en 9. Dado que el recuento actual de montajes es de 8, el próximo arranque del sistema provocará una verificación del sistema de archivos.

2. Considere la siguiente salida de `du -h`:

   ```
   $ du -h
   216K    ./somedir/anotherdir
   224K    ./somedir
   232K    .
   ```

   ¿Cuánto espacio ocupan solo los archivos en el directorio actual? ¿Cómo podríamos reescribir el comando para mostrar esta información con mayor claridad?

   Del total de 232K utilizados, 224K son utilizados por el subdirectorio `somedir` y sus subdirectorios. Entonces, excluyendo esos, tenemos 8K ocupados por los archivos en el directorio actual. Esta información se puede mostrar más claramente usando el parámetro `-S`, que separará los directorios en el conteo.

3. ¿Qué pasaría con el sistema de archivos ext2 `/dev/sdb1` si se emite el siguiente comando?

   ```
   # tune2fs -j /dev/sdb1 -J device=/dev/sdc1 -i 30d
   ```

   Se agregará un journal a `/dev/sdb1`, convirtiéndolo en ext3. El journal se almacenará en el dispositivo `/dev/sdc1` y el sistema de archivos se comprobará cada 30 días.

4. ¿Cómo podemos verificar si hay errores en un sistema de archivos XFS en `/dev/sda1` que tiene una sección de registro en `/dev/sdc1`, sin realmente hacer ninguna reparación?

Use `xfs_repair`, seguido de `-l /dev/sdc1` para indicar el dispositivo que contiene la sección de registro, y `-n` para evitar hacer cambios.

```
# xfs_repair -l /dev/sdc1 -n
```

5. ¿Cuál es la diferencia entre los parámetros `-T` y `-t` para `df`?

   El parámetro `-T` incluirá el tipo de cada sistema de archivos en la salida de `df`. `-t` es un filtro y mostrará solo los sistemas de archivos del tipo dado en la salida, excluyendo todos los demás.



104.3 Controlar el montaje y desmontaje de los sistemas de archivos

Referencia al objetivo del LPI

LPIC-1 version 5.0, Exam 101, Objective 104.3

Importancia

3

Áreas de conocimiento clave

- Montar y desmontar sistemas de archivos de forma manual.
- Configurar el montaje del sistema de archivos en el arranque.
- Configurar sistemas de archivos extraibles y montables por el usuario.
- Uso de etiquetas e identificadores únicos universales (UUIDs) para la identificación y el * montaje de sistemas de archivos.
- Conocimientos de las unidades de montaje de systemd.

Lista parcial de archivos, términos y utilidades

- `/etc/fstab`
- `/media/`
- `mount`
- `umount`
- `blkid`
- `lsblk`



104.3 Lección 1

Certificación:	LPIC-1
Versión:	5.0
Tema:	104 Dispositivos, sistemas de archivos Linux, estándar de jerarquía del sistema de archivos
Objetivo:	104.3 Controlar el montaje y desmontaje de sistemas de archivos
Lección:	1 de 1

Introducción

Hasta ahora, aprendió cómo particionar discos y cómo crear y mantener sistemas de archivos en ellos. Sin embargo, antes de poder acceder a un sistema de archivos en Linux, es necesario *montarlo*.

Esto significa adjuntar el sistema de archivos a un punto específico en el árbol de directorios de su sistema, llamado *punto de montaje*. Los sistemas de archivos se pueden montar manual o automáticamente y hay muchas formas de hacerlo. Aprenderemos sobre algunos de ellos en esta lección.

Montaje y desmontaje de sistemas de archivos

El comando para montar manualmente un sistema de archivos se llama `mount` y su sintaxis es:

```
mount -t TYPE DEVICE MOUNTPOINT
```

Donde:

`TYPE`
El tipo de sistema de archivos que se está montando (por ejemplo, ext4, btrfs, exfat, etc.).

`DEVICE`
El nombre de la partición que contiene el sistema de archivos (por ejemplo, `/dev/sdb1`)

`MOUNTPOINT`
Dónde se montará el sistema de archivos. No es necesario que el directorio en el que esté montado esté vacío, aunque debe existir. Sin embargo, cualquier archivo que contenga será inaccesible por su nombre mientras el sistema de archivos esté montado.

Por ejemplo, para montar una unidad flash USB que contenga un sistema de archivos exFAT ubicado en `/dev/sdb1` en un directorio llamado `flash` en su directorio de inicio, puede usar:

```
# mount -t exfat /dev/sdb1 ~/flash/
```

TIP Muchos sistemas Linux usan el shell Bash, y en ellos la tilde `~` en la ruta al punto de montaje es una abreviatura del directorio de inicio del usuario actual. Si el usuario actual se llama `john`, por ejemplo, será reemplazado por `/home/john`.

Después del montaje, se podrá acceder al contenido del sistema de archivos en el directorio `~/flash`:

```
$  ls -lh ~/flash/
total 469M
-rwxrwxrwx 1 root root 454M jul 19 09:49 lineage-16.0-20190711-MOD-quark.zip
-rwxrwxrwx 1 root root  16M jul 19 09:44 twrp-3.2.3-mod_4-quark.img
```

Listado de sistemas de archivos montados

Si teclea simplemente `mount`, obtendrá una lista de todos los sistemas de archivos actualmente montados en su sistema. Esta lista puede ser bastante grande porque, además de los discos conectados a su sistema, también contiene varios sistemas de archivos en tiempo de ejecución en la memoria que sirven para varios propósitos. Para filtrar la salida, puede usar el parámetro `-t` para listar solo los sistemas de archivos del tipo correspondiente, como se muestra a continuación:

```
# mount -t ext4
/dev/sda1 on / type ext4 (rw,noatime,errors=remount-ro)
```

Puede especificar varios sistemas de archivos a la vez separándolos con una coma:

```
# mount -t ext4,fuseblk
/dev/sda1 on / type ext4 (rw,noatime,errors=remount-ro)
/dev/sdb1 on /home/carol/flash type fuseblk
(rw,nosuid,nodev,relatime,user_id=0,group_id=0,default_permissions,allow_other,blks
ize=4096) [DT_8GB]
```

La salida de los ejemplos anteriores se puede describir en el formato:

```
SOURCE on TARGET type TYPE OPTIONS
```

Donde `SOURCE` es la partición que contiene el sistema de archivos, `TARGET` es el directorio donde está montado, `TYPE` es el tipo de sistema de archivos y `OPTIONS` son las opciones pasadas al comando `mount` en el momento del montaje.

Parámetros adicionales de la línea de comandos

Hay muchos parámetros de línea de comandos que se pueden usar con `mount`. Algunas de las más utilizadas son:

`-a`

Esto montará todos los sistemas de archivos listados en el archivo `/etc/fstab` (más sobre eso en la siguiente sección).

`-o` o **`--options`**

Esto pasará una lista de *opciones de montaje* separadas por comas al comando de montaje, que puede cambiar cómo se montará el sistema de archivos. Estos también se discutirán junto con `/etc/fstab`.

`-r` o **`-ro`**

Esto montará el sistema de archivos como de solo lectura.

`-w` o **`-rw`**

Esto hará que el sistema de archivos de montaje sea escribible.

Para desmontar un sistema de archivos, use el comando `umount`, seguido del nombre del dispositivo o el punto de montaje. Teniendo en cuenta el ejemplo anterior, los comandos siguientes son intercambiables:

```
# umount /dev/sdb1
# umount ~/flash
```

Algunos de los parámetros de la línea de comandos para `desmontar` son:

`-a`
: Esto desmontará todos los sistemas de archivos listados en `/etc/fstab`.

`-f`
: Esto forzará el desmontaje de un sistema de archivos. Esto puede resultar útil si ha montado un sistema de archivos remoto que se ha vuelto inalcanzable.

`-r`
: Si el sistema de archivos no se puede desmontar, esto intentará convertirlo en solo lectura.

Tratamiento de archivos abiertos

Al desmontar un sistema de archivos, puede encontrar un mensaje de error que indique que el `target is busy`. Esto sucederá si hay archivos abiertos en el sistema de archivos. Sin embargo, puede que no sea obvio de inmediato dónde se encuentra un archivo abierto o qué está accediendo al sistema de archivos.

En tales casos, puede usar el comando `lsof`, seguido del nombre del dispositivo que contiene el sistema de archivos, para ver una lista de los procesos que acceden a él y qué archivos están abiertos. Por ejemplo:

```
# umount /dev/sdb1
umount: /media/carol/External_Drive: target is busy.

# lsof /dev/sdb1
COMMAND  PID   USER   FD   TYPE DEVICE SIZE/OFF NODE NAME
evince  3135 carol   16r   REG   8,17 21881768 5195
/media/carol/External_Drive/Documents/E-Books/MagPi40.pdf
```

`COMMAND` es el nombre del ejecutable que abrió el archivo y `PID` es el número de proceso. `NAME` es el nombre del archivo que está abierto. En el ejemplo anterior, el archivo `MagPi40.pdf` es abierto por el programa `evince` (un visor de PDF). Si cerramos el programa, podremos desmontar el sistema de archivos.

NOTE

Antes de que aparezca la salida `lsof`, los usuarios de `GNOME` pueden ver un mensaje de advertencia en la ventana del terminal.

```
lsof: WARNING: can't stat() fuse.gvfsd-fuse file system
/run/user/1000/gvfs
Output information may be incomplete.
```

`lsof` intenta procesar todos los sistemas de archivos montados. Este mensaje de advertencia se genera porque `lsof` ha encontrado un sistema de archivos virtual `GNOME` (GVFS). Este es un caso especial de un sistema de archivos en el espacio de usuario (`FUSE`). Actúa como un puente entre `GNOME`, sus API y el kernel. Nadie, ni siquiera root, puede acceder a uno de estos sistemas de archivos, aparte del propietario que lo montó (en este caso, `GNOME`). Puede ignorar esta advertencia.

¿Dónde montar?

Puede montar un sistema de archivos en cualquier lugar que desee. Sin embargo, hay algunas buenas prácticas que deben seguirse para facilitar la administración del sistema.

Tradicionalmente, `/mnt` era el directorio en el que se montarían todos los dispositivos externos y una serie de "puntos de anclaje" preconfigurados para dispositivos comunes, como unidades de CD-ROM (`/mnt/cdrom`) y disquetes. (`/mnt/floppy`) existían en él.

Esto ha sido reemplazado por `/media`, que ahora es el punto de montaje predeterminado para cualquier medio extraíble por el usuario (por ejemplo, discos externos, unidades flash USB, lectores de tarjetas de memoria, etc.) conectados al sistema.

En la mayoría de las distribuciones modernas de Linux y entornos de escritorio, los dispositivos extraíbles se montan automáticamente en `/media/USER/LABEL` cuando se conectan al sistema, donde `USER` es el nombre de usuario y `LABEL` es la etiqueta del dispositivo. Por ejemplo, una unidad flash USB con la etiqueta `FlashDrive` conectada por el usuario `john` se montaría en `/media/john/FlashDrive/`. La forma en que se maneja esto es diferente según el entorno de escritorio.

Dicho esto, siempre que necesite montar *manualmente* un sistema de archivos, es una buena práctica montarlo en `/mnt`.

Montaje de sistemas de archivos en el arranque

El archivo `/etc/fstab` contiene descripciones sobre los sistemas de archivos que se pueden montar.

Este es un archivo de texto, donde cada línea describe un sistema de archivos que se va a montar, con seis campos por línea en el siguiente orden:

```
FILESYSTEM MOUNTPOINT TYPE OPTIONS DUMP PASS
```

Donde:

`FILESYSTEM`
: El dispositivo que contiene el sistema de archivos que se va a montar. En lugar del dispositivo, puede especificar el UUID o etiqueta de la partición, algo que discutiremos más adelante.

`MOUNTPOINT`
: Dónde se montará el sistema de archivos.

`TYPE`
: El tipo de sistema de archivos.

`OPTIONS`
: Opciones de montaje que se pasarán a `mount`.

`DUMP`
: Indica si cualquier sistema de archivos ext2, ext3 o ext4 debe considerarse para la copia de seguridad mediante el comando `dump`. Por lo general, es cero, lo que significa que deben ignorarse.

`PASS`
: Cuando es distinto de cero, define el orden en el que se comprobarán los sistemas de archivos durante el arranque. Normalmente es cero.

Por ejemplo, la primera partición en el primer disco de una máquina podría describirse como:

```
/dev/sda1  /  ext4  noatime,errors
```

Las opciones de montaje en `OPTIONS` son una lista de parámetros separados por comas, que pueden ser genéricos o específicos del sistema de archivos. Entre los genéricos tenemos:

`atime` y **`noatime`**
: Por defecto, cada vez que se lee un archivo, se actualiza la información de tiempo de acceso.

Deshabilitar esto (con `noatime`) puede acelerar la E/S del disco. No confunda esto con la hora de modificación, que se actualiza cada vez que se escribe un archivo.

`auto` y `noauto`
Si el sistema de archivos puede (o no) montarse automáticamente con `mount -a`.

`defaults`
Esto pasará las opciones `rw`, `suid`, `dev`, `exec`, `auto`, `nouser` y `async` a `mount`.

`dev` y `nodev`
Si deben interpretarse los dispositivos de caracteres o bloques en el sistema de archivos montado.

`exec` y `noexec`
Permitir o denegar el permiso para ejecutar binarios en el sistema de archivos.

`user` y `nouser`
Permite (o no) a un usuario normal montar el sistema de archivos.

`group`
Permite a un usuario montar el sistema de archivos si el usuario pertenece al mismo grupo que posee el dispositivo que lo contiene.

`owner`
Permite a un usuario montar un sistema de archivos si el usuario posee el dispositivo que lo contiene.

`suid` y `nosuid`
Permite, o no, que los bits SETUID y SETGID surtan efecto.

`ro` y `rw`
Montan un sistema de archivos como de solo lectura o de escritura.

`remount`
Esto intentará volver a montar un sistema de archivos ya montado. Esto no se usa en `/etc/fstab`, sino como un parámetro para `mount -o`. Por ejemplo, para volver a montar la partición `/dev/sdb1` ya montada como de solo lectura, puede usar el comando `mount -o remount,ro /dev/sdb1`. Al volver a montar, no es necesario especificar el tipo de sistema de archivos, solo el nombre del dispositivo *o* el punto de montaje.

`sync` y `async`

Realizar todas las operaciones de E/S en el sistema de archivos de forma sincrónica o asincrónica. `async` suele ser el predeterminado. La página del manual de `mount` advierte que el uso de `sync` en medios con un número limitado de ciclos de escritura (como unidades flash o tarjetas de memoria) puede acortar la vida útil del dispositivo.

Uso de UUID y etiquetas

Especificar el nombre del dispositivo que contiene el sistema de archivos a montar puede presentar algunos problemas. A veces, el mismo nombre de dispositivo puede asignarse a otro dispositivo dependiendo de cuándo o dónde se conectó a su sistema. Por ejemplo, una unidad flash USB en `/dev/sdb1` puede asignarse a `/dev/sdc1` si se conecta a otro puerto, o después de otra unidad flash.

Una forma de evitar esto es especificar la etiqueta o UUID (*Universally Unique Identifier*) del volumen. Ambos se especifican cuando se crea el sistema de archivos y no cambiarán, a menos que el sistema de archivos se destruya o se le asigne manualmente una nueva etiqueta o UUID.

El comando `lsblk` se puede utilizar para consultar información sobre un sistema de archivos y averiguar la etiqueta y el UUID asociados a él. Para hacer esto, use el parámetro `-f`, seguido del nombre del dispositivo:

```
$ lsblk -f /dev/sda1
NAME FSTYPE LABEL UUID                                 FSAVAIL FSUSE% MOUNTPOINT
sda1 ext4         6e2c12e3-472d-4bac-a257-c49ac07f3761   64,9G    33% /
```

Aquí está el significado de cada columna:

`NAME`

Nombre del dispositivo que contiene el sistema de archivos.

`FSTYPE`

Tipo de sistema de archivos.

`LABEL`

Etiqueta del sistema de archivos.

`UUID`

Identificador único universal (UUID) asignado al sistema de archivos.

`FSAVAIL`
: Cuánto espacio hay disponible en el sistema de archivos.

`FSUSE%`
: Porcentaje de uso del sistema de archivos.

`MOUNTPOINT`
: Dónde está montado el sistema de archivos.

En `/etc/fstab` un dispositivo se puede especificar por su UUID con la opción `UUID=`, seguido del UUID, o con `LABEL=`, seguido de la etiqueta. Entonces, en lugar de:

```
/dev/sda1  /  ext4  noatime,errors
```

Usarías:

```
UUID=6e2c12e3-472d-4bac-a257-c49ac07f3761  /  ext4  noatime,errors
```

O, si tiene un disco con la etiqueta `homedisk`:

```
LABEL=homedisk  /home ext4  defaults
```

La misma sintaxis se puede utilizar con el comando `mount`. En lugar del nombre del dispositivo, pase el UUID o la etiqueta. Por ejemplo, para montar un disco NTFS externo con el UUID `56C11DCC5D2E1334` en `/mnt/external`, el comando sería:

```
$ mount -t ntfs UUID=56C11DCC5D2E1334 /mnt/external
```

Montaje de discos con Systemd

Systemd es el *init* del sistema, el primer proceso que se ejecuta en muchas distribuciones de Linux. Es responsable de generar otros procesos, iniciar servicios y arrancar el sistema. Entre muchas otras tareas, systemd también se puede utilizar para gestionar el montaje (y montaje automático) de sistemas de archivos.

Para utilizar esta función de systemd, debe crear un archivo de configuración llamado *mount unit*. Cada volumen que se va a montar tiene su propia unidad de montaje y es necesario colocarlos en

`/etc/systemd/system/`.

Las unidades de montaje son archivos de texto simples con la extensión `.mount`. El formato básico se muestra a continuación:

```
[Unit]
Description=

[Mount]
What=
Where=
Type=
Options=

[Install]
WantedBy=
```

`Description=`
: Breve descripción de la unidad de montaje, algo así como `Monta el disco de respaldo`.

`What=`
: Qué se debe montar. El volumen debe especificarse como `/dev/disk/by-uuid/VOL_UUID` donde `VOL_UUID` es el UUID del volumen.

`Where=`
: Debe ser la ruta completa hacia donde se debe montar el volumen.

`Type=`
: El tipo de sistema de archivos.

`Options=`
: Las opciones de montaje que desee pasar, son las mismas que se utilizan con el comando `mount` o en `/etc/fstab`.

`WantedBy=`
: Se utiliza para la gestión de dependencias. En este caso, usaremos `multi-user.target`, lo que significa que siempre que el sistema se inicie en un entorno multiusuario (un inicio normal), se montará la unidad.

Nuestro ejemplo anterior del disco externo podría escribirse como:

```
[Unit]
Description=External data disk

[Mount]
What=/dev/disk/by-uuid/56C11DCC5D2E1334
Where=/mnt/external
Type=ntfs
Options=defaults

[Install]
WantedBy=multi-user.target
```

Pero aún no hemos terminado. Para que funcione correctamente, la unidad de montaje *debe* tener el mismo nombre que el punto de montaje. En este caso, el punto de montaje es `/mnt/external`, por lo que el archivo debe llamarse `mnt-external.mount`.

Después de eso, debe reiniciar el demonio systemd con el comando `systemctl` e iniciar la unidad:

```
# systemctl daemon-reload
# systemctl start mnt-external.mount
```

Ahora el contenido del disco externo debería estar disponible en `/mnt/external`. Puede verificar el estado del montaje con el comando `systemctl status mnt-external.mount`, como a continuación:

```
# systemctl status mnt-external.mount
▯ mnt-external.mount - External data disk
   Loaded: loaded (/etc/systemd/system/mnt-external.mount; disabled; vendor pres
   Active: active (mounted) since Mon 2019-08-19 22:27:02 -03; 14s ago
    Where: /mnt/external
     What: /dev/sdb1
    Tasks: 0 (limit: 4915)
   Memory: 128.0K
   CGroup: /system.slice/mnt-external.mount

ago 19 22:27:02 pop-os systemd[1]: Mounting External data disk...
ago 19 22:27:02 pop-os systemd[1]: Mounted External data disk.
```

El comando `systemctl start mnt-external.mount` solo habilitará la unidad para la sesión actual. Si desea habilitarlo en cada arranque, reemplace `start` por `enable`:

```
# systemctl enable mnt-external.mount
```

Montaje automático de una unidad de montaje

Las unidades de montaje se pueden montar automáticamente siempre que se acceda al punto de montaje. Para hacer esto, necesita un archivo `.automount`, junto con el archivo `.mount` que describe la unidad. El formato básico es:

```
[Unit]
Description=

[Automount]
Where=

[Install]
WantedBy=multi-user.target
```

Como antes, `Description=` es una breve descripción del archivo y `Where=` es el punto de montaje. Por ejemplo, un archivo `.automount` para nuestro ejemplo anterior sería:

```
[Unit]
Description=Automount for the external data disk

[Automount]
Where=/mnt/external

[Install]
WantedBy=multi-user.target
```

Guarde el archivo con el mismo nombre que el punto de montaje (en este caso, `mnt-external.automount`), vuelva a cargar systemd e inicie la unidad:

```
# systemctl daemon-reload
# systemctl start mnt-external.automount
```

Ahora, siempre que se acceda al directorio `/mnt/external`, se montará el disco. Como antes, para habilitar el montaje automático en cada arranque, usaría:

```
# systemctl enable mnt-external.automount
```

Ejercicios Guiados

1. Usando `mount`, ¿cómo se puede montar un sistema de archivos `ext4` en `/dev/sdc1` a `/mnt/external` como solo lectura, usando las opciones `noatime` y `async`?

2. Al desmontar un sistema de archivos en `/dev/sdd2`, aparece el mensaje de error `target is busy`. ¿Cómo puede saber qué archivos del sistema de archivos están abiertos y qué procesos los abrieron?

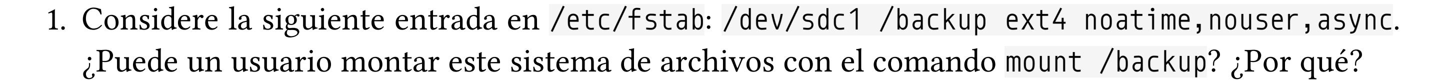
3. Considere la siguiente entrada en `/etc/fstab`: `/dev/sdb1 /data ext4 noatime,noauto,async`. ¿Se montará este sistema de archivos si se emite el comando `mount -a`? ¿Por qué?

4. ¿Cómo se puede averiguar el UUID de un sistema de archivos en `/dev/sdb1`?

5. ¿Cómo se puede usar `mount` para volver a montar como de solo lectura un sistema de archivos exFAT con el UUID `6e2c12e3-472d-4bac-a257-c49ac07f3761`, montado en `/mnt/data`?

6. ¿Cómo se puede obtener una lista de todos los sistemas de archivos `ext3` y `ntfs` montados actualmente en un sistema?

Ejercicios Exploratorios



1. Considere la siguiente entrada en `/etc/fstab`: `/dev/sdc1 /backup ext4 noatime,nouser,async`. ¿Puede un usuario montar este sistema de archivos con el comando `mount /backup`? ¿Por qué?

2. Considere un sistema de archivos remoto montado en `/mnt/server`, que se ha vuelto inalcanzable debido a una pérdida de conectividad de red. ¿Cómo podría obligarlo a desmontarlo o montarlo como de solo lectura si esto no es posible?

3. Escriba una entrada `/etc/fstab` que monte un volumen btrfs con la etiqueta `Backup` en `/mnt/backup`, con opciones predeterminadas y sin permitir la ejecución de binarios desde él.

4. Considere la siguiente unidad de montaje systemd:

```
[Unit]
Description=External data disk

[Mount]
What=/dev/disk/by-uuid/56C11DCC5D2E1334
Where=/mnt/external
Type=ntfs
Options=defaults

[Install]
WantedBy=multi-user.target
```

   - ¿Cuál sería una entrada `/etc/fstab` equivalente para este sistema de archivos?

5. ¿Cuál debería ser el nombre de archivo de la unidad anterior para que pueda ser utilizada por systemd? ¿Dónde debería estar?

Resumen

En esta lección, aprendió a montar y desmontar sistemas de archivos, de forma manual o automática. Algunos de los comandos y conceptos explicados fueron:

- `mount` (monta un dispositivo en una ubicación)
- `umount` (desmonta un dispositivo)
- `lsof` (lista los procesos que acceden a un sistema de archivos)
- Directorios `/mnt` y `/media`
- `/etc/fstab`
- `lsblk` (lista el tipo y UUID de un sistema de archivos)
- Cómo montar un sistema de archivos usando su UUID o etiqueta.
- Cómo montar un sistema de archivos usando unidades de montaje systemd.
- Cómo montar automáticamente un sistema de archivos utilizando unidades de montaje systemd.

Respuestas a los ejercicios guiados

1. Usando `mount`, ¿cómo se puede montar un sistema de archivos `ext4` en `/dev/sdc1` a `/mnt/external` como solo lectura, usando las opciones `noatime` y `async`?

   ```
   # mount -t ext4 -o noatime,async,ro /dev/sdc1 /mnt/external
   ```

2. Al desmontar un sistema de archivos en `/dev/sdd2`, aparece el mensaje de error `target is busy`. ¿Cómo puede saber qué archivos del sistema de archivos están abiertos y qué procesos los abrieron?

   Utilice `lsof` seguido del nombre del dispositivo:

   ```
   $ lsof /dev/sdd2
   ```

3. Considere la siguiente entrada en `/etc/fstab`: `/dev/sdb1 /data ext4 noatime,noauto,async`. ¿Se montará este sistema de archivos si se emite el comando `mount -a`? ¿Por qué?

   No se montará. La clave es el parámetro `noauto`, lo que significa que `mount -a` ignorará esta entrada.

4. ¿Cómo se puede averiguar el UUID de un sistema de archivos en `/dev/sdb1`?

   Utilice `lsblk -f`, seguido del nombre del sistema de archivos:

   ```
   $ lsblk -f /dev/sdb1
   ```

5. ¿Cómo se puede usar `mount` para volver a montar como de solo lectura un sistema de archivos exFAT con el UUID `6e2c12e3-472d-4bac-a257-c49ac07f3761`, montado en `/mnt/data`?

   Dado que el sistema de archivos está montado, no necesita preocuparse por el tipo de sistema de archivos o el ID, simplemente use la opción `remount` con el parámetro `ro` (solo lectura) y el punto de montaje:

   ```
   # mount -o remount,ro /mnt/data
   ```

6. ¿Cómo se puede obtener una lista de todos los sistemas de archivos `ext3` y `ntfs` montados actualmente en un sistema?

Utilice `mount -t`, seguido de una lista de sistemas de archivos separados por comas:

```
# mount -t ext3,ntfs
```

Respuestas a ejercicios exploratorios

1. Considere la siguiente entrada en `/etc/fstab`: `/dev/sdc1 /backup ext4 noatime,nouser,async`. ¿Puede un usuario montar este sistema de archivos con el comando `mount /backup`? ¿Por qué?

   No, el parámetro `nouser` no permitirá a los usuarios normales montar este sistema de archivos.

2. Considere un sistema de archivos remoto montado en `/mnt/server`, que se ha vuelto inalcanzable debido a una pérdida de conectividad de red. ¿Cómo podría obligarlo a desmontarlo o montarlo como de solo lectura si esto no es posible?

   Pase los parámetros `-f` y `-r` para desmontar. El comando sería `umount -f -r /mnt/server`. Recuerde que puede agrupar parámetros, por lo que `umount -fr /mnt/server` también funcionaría.

3. Escriba una entrada `/etc/fstab` que monte un volumen btrfs con la etiqueta `Backup` en `/mnt/backup`, con opciones predeterminadas y sin permitir la ejecución de binarios desde él.

   La línea debe ser `LABEL=Backup /mnt/backup btrfs defaults,noexec`

4. Considere la siguiente unidad de montaje systemd:

```
[Unit]
Description=External data disk

[Mount]
What=/dev/disk/by-uuid/56C11DCC5D2E1334
Where=/mnt/external
Type=ntfs
Options=defaults

[Install]
WantedBy=multi-user.target
```

   - ¿Cuál sería una entrada `/etc/fstab` equivalente para este sistema de archivos?

     La entrada sería: `UUID=56C11DCC5D2E1334 /mnt/external ntfs defaults`

5. ¿Cuál debería ser el nombre de archivo de la unidad anterior para que pueda ser utilizada por systemd? ¿Dónde debería estar?

   El nombre del archivo debe ser el mismo que el del punto de montaje, por lo que `mnt-`

`external.mount`, colocado en `/etc/systemd/system`.



104.5 Administración de los permisos y los propietarios de los archivos

Referencia al objetivo del LPI

LPIC-1 version 5.0, Exam 101, Objective 104.5

Importancia

3

Áreas de conocimiento clave

- Administrar los permisos de acceso a archivos regulares y especiales así como a directorios.
- Usar modos de acceso tales como el suid, el sgid y el sticky bit para mantener la seguridad.
- Saber cambiar la máscara de creación de archivos.
- Usar el campo grupo para otorgar acceso a archivos a miembros de un grupo.

Lista parcial de archivos, términos y utilidades

- `chmod`
- `umask`
- `chown`
- `chgrp`



104.5 Lección 1

Certificación:	LPIC-1
Versión:	5.0
Tema:	104 Dispositivos, sistemas de archivos Linux, estándar de jerarquía del sistema de archivos
Objetivo:	104.5 Administrar los permisos y la propiedad de los archivos
Lección:	1 de 1

Introducción

Al ser un sistema multiusuario, Linux necesita alguna forma de rastrear quién es el propietario de cada archivo y si un usuario puede o no realizar acciones en un archivo. Esto es para garantizar la privacidad de los usuarios que deseen mantener la confidencialidad del contenido de sus archivos, así como para garantizar la colaboración al hacer que ciertos archivos sean accesibles para múltiples usuarios.

Esto se realiza mediante un sistema de permisos de tres niveles. Cada archivo en el disco es propiedad de un usuario y un grupo de usuarios y tiene tres conjuntos de permisos: uno para su propietario, otro para el grupo propietario del archivo y otro para todos los demás. En esta lección, aprenderá a consultar los permisos de un archivo, el significado de estos permisos y cómo manipularlos.

Consultar información sobre archivos y directorios

El comando `ls` se usa para obtener una lista del contenido de cualquier directorio. En esta forma

básica, todo lo que obtiene son los nombres de archivo:

```
$ ls
Another_Directory  picture.jpg  text.txt
```

Pero hay mucha más información disponible para cada archivo, incluido su tipo, tamaño, propiedad y más. Para ver esta información debe solicitar a `ls` una lista de "formato largo", usando el parámetro `-l`:

```
$ ls -l
total 536
drwxrwxr-x 2 carol carol   4096 Dec 10 15:57 Another_Directory
-rw------- 1 carol carol 539663 Dec 10 10:43 picture.jpg
-rw-rw-r-- 1 carol carol   1881 Dec 10 15:57 text.txt
```

Cada columna de la salida anterior tiene un significado. Echemos un vistazo a las columnas relevantes para esta lección.

- La *primera* columna de la lista muestra el tipo de archivo y los permisos. Por ejemplo, en `drwxrwxr-x`:
  - El primer caracter, `d`, indica el tipo de archivo.
  - Los siguientes tres caracteres, `rwx`, indican los permisos del propietario del archivo, también conocido como *user* o `u`.
  - Los siguientes tres caracteres, `rwx`, indican los permisos del *grupo* propietario del archivo, también denominado `g`.
  - Los últimos tres caracteres, `r-x`, indican los permisos para cualquier otra persona, también conocidos como *otros* u `o`.

TIP También es común escuchar el conjunto de permisos de *otros* como permisos de *mundo*, como en "Todo el mundo tiene estos permisos".

- Las columnas *tercera* y *cuarta* muestran información de propiedad: respectivamente, el usuario y el grupo que posee el archivo.
- La *séptima* y última columna muestra el nombre del archivo.

La *segunda* columna indica el número de enlaces físicos que apuntan a ese archivo. La *quinta* columna muestra el tamaño del archivo. La *sexta* columna muestra la fecha y la hora en que se

modificó por última vez el archivo. Pero estas columnas no son relevantes para el tema actual.

¿Y los directorios?

Si intenta consultar información sobre un directorio usando `ls -l`, en su lugar, le mostrará una lista del contenido del directorio:

```
$ ls -l Another_Directory/
total 0
-rw-r--r-- 1 carol carol 0 Dec 10 17:59 another_file.txt
```

Para evitar esto y consultar información sobre el directorio en sí, agregue el parámetro `-d` a `ls`:

```
$ ls -l -d Another_Directory/
drwxrwxr-x 2 carol carol 4096 Dec 10 17:59 Another_Directory/
```

Ver archivos ocultos

El listado de directorios que hemos recuperado usando `ls -l` antes está incompleto:

```
$ ls -l
total 544
drwxrwxr-x 2 carol carol   4096 Dec 10 17:59 Another_Directory
-rw------- 1 carol carol 539663 Dec 10 10:43 picture.jpg
-rw-rw-r-- 1 carol carol   1881 Dec 10 15:57 text.txt
```

Hay otros tres archivos en ese directorio, pero están ocultos. En Linux, los archivos cuyo nombre comienzan con un punto (`.`) se ocultan automáticamente. Para verlos necesitamos agregar el parámetro `-a` a `ls`:

```
$ ls -l -a
total 544
drwxrwxr-x 3 carol carol   4096 Dec 10 16:01 .
drwxrwxr-x 4 carol carol   4096 Dec 10 15:56 ..
drwxrwxr-x 2 carol carol   4096 Dec 10 17:59 Another_Directory
-rw------- 1 carol carol 539663 Dec 10 10:43 picture.jpg
-rw-rw-r-- 1 carol carol   1881 Dec 10 15:57 text.txt
-rw-r--r-- 1 carol carol      0 Dec 10 16:01 .thisIsHidden
```

El archivo `.thisIsHidden` simplemente está oculto porque su nombre comienza con `.`.

Sin embargo, los directorios `.` y `..` son especiales. `.` es un puntero al directorio padre. Y `..` es un puntero al directorio principal, el que contiene el actual. En Linux, cada directorio contiene al menos estos dos directorios.

TIP Puede combinar varios parámetros para `ls` (y muchos otros comandos de Linux). `ls -l -a` puede, por ejemplo, escribirse como `ls -la`.

Tipos de Archivos

Ya hemos mencionado que la primera letra en cada salida de `ls -l` describe el tipo de archivo. Los tres tipos de archivos más comunes son:

`-` (archivo normal)

Un archivo puede contener datos de cualquier tipo y ayudar a gestionar estos datos. Los archivos se pueden modificar, mover, copiar y eliminar.

`d` (directorio)

Un directorio contiene otros archivos o directorios y ayuda a organizar el sistema de archivos. Técnicamente, los directorios son un tipo especial de archivo.

`l` (enlace simbólico)

Este “archivo” es un puntero a otro archivo o directorio en otra ubicación del sistema de archivos.

Además de estos, hay otros tres tipos de archivos que al menos debería conocer, pero que están fuera del alcance de esta lección:

`b` (dispositivo de bloque)

Este archivo representa un dispositivo virtual o físico, generalmente discos u otros tipos de dispositivos de almacenamiento, como el primer disco duro que podría estar representado por `/dev/sda`.

`c` (dispositivo de caracteres)

Este archivo representa un dispositivo físico o virtual. Los terminales (como el terminal principal en `/dev/ttyS0`) y los puertos serie son ejemplos comunes de dispositivos de caracteres.

`s` (socket)

Los sockets sirven como “conductos” para pasar información entre dos programas.

WARNING No altere ninguno de los permisos en dispositivos de bloque, dispositivos de caracteres o sockets, a menos que sepa lo que está haciendo. ¡Esto puede impedir que su sistema funcione!

Comprensión de los Permisos

En la salida de `ls -l`, los permisos de archivo se muestran justo después del tipo de archivo, como tres grupos de tres caracteres cada uno, en el orden `r`, `w` y `x`. Esto es lo que quieren decir. Tenga en cuenta que un guión `-` representa la falta de un permiso.

Permisos sobre archivos

`r`

Significa *read* y tiene un valor octal de `4` (no se preocupe, discutiremos los octales en breve). Esto significa permiso para abrir un archivo y leer su contenido.

`w`

Significa *write* y tiene un valor octal de `2`. Esto significa permiso para editar o eliminar un archivo.

`x`

Significa *execute* y tiene un valor octal de `1`. Esto significa que el archivo se puede ejecutar como ejecutable o script.

Entonces, por ejemplo, un archivo con permisos `rw-` se puede leer y escribir, pero no se puede ejecutar.

Permisos en directorios

`r`

Significa *read* y tiene un valor octal de `4`. Esto significa permiso para leer el contenido del directorio, como nombres de archivos. Pero *no* implica permiso para leer los archivos.

`w`

Significa *write* y tiene un valor octal de `2`. Esto significa permiso para crear o eliminar archivos en un directorio, o cambiar sus nombres, permisos y propietarios.

Si un usuario tiene el permiso `w` en un directorio, el usuario puede cambiar los permisos de cualquier archivo en el directorio (el *contenido* del directorio), incluso si el usuario no tiene permisos sobre el archivo o si el archivo es propiedad de otro usuario.

Tenga en cuenta que tener permisos de escritura en un directorio o archivo no significa que tenga permiso para eliminar o cambiar el nombre del directorio o archivo.

`x`

Significa *execute* y tiene un valor octal de `1`. Esto significa permiso para ingresar a un directorio, pero no para listar sus archivos (para eso se necesita `r`).

Lo último sobre directorios puede sonar un poco confuso. Imaginemos, por ejemplo, que tiene un directorio llamado `Another_Directory`, con los siguientes permisos:

```
$ ls -ld Another_Directory/
d--x--x--x 2 carol carol 4,0K Dec 20 18:46 Another_Directory
```

También imagine que dentro de este directorio tiene un script de shell llamado `hello.sh`:

```
-rwxr-xr-x 1 carol carol 33 Dec 20 18:46 hello.sh
```

Si usted es el usuario `carol` e intenta listar el contenido de `Another_Directory`, obtendrá un mensaje de error, ya que su usuario no tiene permiso de lectura para ese directorio:

```
$ ls -l Another_Directory/
ls: cannot open directory 'Another_Directory/': Permission denied
```

Sin embargo, el usuario `carol` *tiene* permisos de ejecución, lo que significa que puede ingresar al directorio. Por lo tanto, el usuario `carol` puede acceder a archivos dentro del directorio, siempre que tenga los permisos correctos *para el archivo respectivo*. Supongamos que el usuario tiene permisos completos (`rwx`) para el script `hello.sh`. Entonces ella *puede* ejecutar el script, aunque ella *no puede* leer el contenido del directorio que lo contiene si conoce el nombre completo del archivo:

```
$ sh Another_Directory/hello.sh
Hello LPI World!
```

Como dijimos antes, los permisos se especifican en secuencia: primero para el propietario del archivo, luego para el grupo propietario y luego para otros usuarios. Siempre que alguien intenta realizar una acción en el archivo, los permisos se verifican de la misma manera.

Primero, el sistema verifica si el usuario actual es el propietario del archivo y, si esto es cierto, solo aplica el primer conjunto de permisos. De lo contrario, comprueba si el usuario actual pertenece al

grupo propietario del archivo. En ese caso, solo aplica el segundo conjunto de permisos. En cualquier otro caso, el sistema aplicará el tercer conjunto de permisos.

Esto significa que si el usuario actual es el propietario del archivo, solo los permisos de propietario son efectivos, incluso si el grupo u otros permisos son más permisivos que los permisos del propietario.

Modificación de permisos de archivos

El comando `chmod` se usa para modificar los permisos de un archivo y toma al menos dos parámetros: el primero describe qué permisos cambiar y el segundo apunta al archivo o directorio donde se realizará el cambio. Tenga en cuenta que solo el propietario del archivo o el administrador del sistema (root) pueden cambiar los permisos de un archivo.

Los permisos para cambiar se pueden describir de dos formas diferentes, o “modos”.

El primero, llamado *modo simbólico* ofrece un control detallado, lo que le permite agregar o revocar un solo permiso sin modificar otros en el conjunto. El otro modo, llamado *modo octal*, es más fácil de recordar y más rápido de usar si desea establecer todos los valores de permisos a la vez.

Ambos modos conducirán al mismo resultado final. Entonces, por ejemplo, los comandos:

```
$ chmod ug+rw-x,o-rwx text.txt
```

y

```
$ chmod 660 text.txt
```

producirá exactamente la misma salida, un archivo con los permisos establecidos:

```
-rw-rw---- 1 carol carol  765 Dec 20 21:25 text.txt
```

Ahora, veamos cómo funciona cada modo.

Modo simbólico

Al describir qué permisos cambiar en *modo simbólico*, los primeros caracteres indican los permisos que modificará: los del usuario (`u`), del grupo (`g`), de los demás (`o`) y / o para todos (`a`).

Luego debe decirle al comando qué hacer: puede otorgar un permiso (`+`), revocar un permiso (`-`) o establecerlo en un valor específico (`=`).

Por último, especifique sobre qué permiso desea actuar: leer (`r`), escribir (`w`) o ejecutar (`x`).

Por ejemplo, imagina que tenemos un archivo llamado `text.txt` con el siguiente conjunto de permisos:

```
$ ls -l text.txt
-rw-r--r-- 1 carol carol 765 Dec 20 21:25 text.txt
```

Si desea otorgar permisos de escritura a los miembros del grupo que posee el archivo, debe usar el parámetro `g+w`. Es más fácil si lo piensa de esta manera: "Para el grupo (`g`), conceder (`+`) permisos de escritura (`w`)". Entonces, el comando sería:

```
$ chmod g+w text.txt
```

Comprobemos el resultado con `ls`:

```
$ ls -l text.txt
-rw-rw-r-- 1 carol carol 765 Dec 20 21:25 text.txt
```

¿Desea eliminar los permisos de lectura para el propietario del mismo archivo? Piense en ello como: "Para el usuario (`u`), revocar (`-`) los permisos de lectura (`r`)". Entonces el parámetro es `u-r`, así:

```
$ chmod u-r text.txt
$ ls -l text.txt
--w-rw-r-- 1 carol carol 765 Dec 20 21:25 text.txt
```

¿Qué pasa si queremos establecer los permisos exactamente como `rw-` para todos? Entonces piense en ello como: "Para todo (`a`), establecer exactamente (`=`) leer (`r`), escribir (`w`) y no ejecutar (`-`)". Entonces:

```
$ chmod a=rw- text.txt
$ ls -l text.txt
-rw-rw-rw- 1 carol carol 765 Dec 20 21:25 text.txt
```

Por supuesto, es posible modificar varios permisos al mismo tiempo. En este caso, sepárelos con una coma (`,`):

```
$ chmod u+rwx,g-x text.txt
$ ls -lh text.txt
-rwxrw-rw- 1 carol carol 765 Dec 20 21:25 text.txt
```

El ejemplo anterior se puede leer como: “Para el usuario (`u`), conceder (`+`) permisos de lectura, escritura y ejecución (`rwx`), para el grupo (`g`), revocar (`-`) ejecutar permisos (`x`)”.

Cuando se ejecuta en un directorio, `chmod` modifica solo los permisos del directorio. `chmod` también tiene un modo recursivo, que es útil cuando desee cambiar los permisos para “todos los archivos dentro de un directorio y sus subdirectorios”. Para usar esto, agregue el parámetro `-R` después del nombre del comando, antes de los permisos para cambiar:

```
$ chmod -R u+rwx Another_Directory/
```

Este comando se puede leer como: “Recursivamente (`-R`), para el usuario (`u`), otorgar (`+`) permisos de lectura, escritura y ejecución (`rwx`)”.

WARNING	Tenga cuidado y piénselo dos veces antes de usar el modificador `-R`, ya que es fácil cambiar los permisos en archivos y directorios que no desea cambiar, especialmente en directorios con una gran cantidad de archivos y subdirectorios.

Modo Octal

En *modo octal*, los permisos se especifican de forma diferente: como un valor de tres dígitos en notación octal, un sistema numérico de base 8.

Cada permiso tiene un valor correspondiente, y se especifican en el siguiente orden: primero, leer (`r`), que es `4`, luego, escribir (`w`), que es `2` y el último, ejecutar (`x`), representado por `1`. Si no hay permiso, use el valor cero (`0`). Entonces, un permiso de `rwx` sería `7` (`4+2+1`) y `r-x` sería `5` (`4+0+1`).

El primero de los tres dígitos representa los permisos para el dueño (`u`), el segundo para el grupo (`g`) y el tercero para otros (`o`). Si quisiéramos establecer los permisos para un archivo en `rw-rw----`, el valor octal sería `660`:

```
$ chmod 660 text.txt
$ ls -l text.txt
-rw-rw---- 1 carol carol 765 Dec 20 21:25 text.txt
```

Además de esto, la sintaxis en *modo octal* es la misma que en *modo simbólico*, el primer parámetro representa los permisos que desea cambiar y el segundo parámetro apunta al archivo o directorio donde se realizará el cambio.

TIP Si un valor de permiso es *impar*, ¡el archivo seguramente es ejecutable!

¿Qué sintaxis debería utilizar? Se recomienda el *modo octal* si desea cambiar los permisos a un valor específico, por ejemplo, `640` (`rw- r-- ---`).

El *modo simbólico* es más útil si desea invertir solo un valor específico, independientemente de los permisos actuales para el archivo. Por ejemplo, puede agregar permisos de ejecución para el usuario usando solo `chmod u+x script.sh` sin tener en cuenta, o incluso tocar, los permisos actuales para el grupo y otros.

Modificación de la propiedad del archivo

El comando `chown` se usa para modificar la propiedad de un archivo o directorio. La sintaxis es bastante simple:

```
chown USERNAME:GROUPNAME FILENAME
```

Por ejemplo, verifiquemos un archivo llamado `text.txt`:

```
$ ls -l text.txt
-rw-rw---- 1 carol carol 1881 Dec 10 15:57 text.txt
```

El usuario propietario del archivo es `carol` y el grupo también es `carol`. Ahora, cambiaremos el grupo propietario del archivo a otro grupo, como `estudiantes`:

```
$ chown carol:students text.txt
$ ls -l text.txt
-rw-rw---- 1 carol students 1881 Dec 10 15:57 text.txt
```

Tenga en cuenta que el usuario propietario de un archivo no necesita pertenecer al grupo propietario

de un archivo. En el ejemplo anterior, el usuario `carol` no necesita ser miembro del grupo `estudiantes`.

El conjunto de permisos de usuario o grupo se puede omitir si no desea cambiarlos. Entonces, para cambiar solo el grupo propietario de un archivo, usaría `chown :students text.txt`. Para cambiar solo el usuario, el comando sería `chown carol: text.txt` o simplemente `chown carol text.txt`. Alternativamente, puede usar el comando `chgrp estudiantes text.txt`.

A menos que sea el administrador del sistema (root), no puede cambiar la propiedad de un archivo a otro usuario o grupo al que no pertenece. Si intenta hacer esto, obtendrá el mensaje de error `Operation not permitted`.

Consultar grupos

Antes de cambiar la propiedad de un archivo, puede resultar útil saber qué grupos existen en el sistema, qué usuarios son miembros de un grupo y a qué grupos pertenece un usuario.

Para ver qué grupos existen en su sistema, escriba `getent group`. La salida será similar a esta (la salida se ha abreviado):

```
$ getent group
root:x:0:
daemon:x:1:
bin:x:2:
sys:x:3:
adm:x:4:syslog,rigues
tty:x:5:rigues
disk:x:6:
lp:x:7:
mail:x:8:
news:x:9:
uucp:x:10:rigues
```

Si desea saber a qué grupos pertenece un usuario, agregue el nombre de usuario como parámetro a `grupos`:

```
$ groups carol
carol : carol students cdrom sudo dip plugdev lpadmin sambashare
```

Para hacer lo contrario (ver qué usuarios pertenecen a un grupo) use `groupmems`. El parámetro `-g` especifica el grupo, y `-l` listará todos sus miembros:

```
# groupmems -g cdrom -l
carol
```

TIP `groupmems` solo se puede ejecutar como root, el administrador del sistema. Si actualmente no ha iniciado sesión como root, agregue `sudo` antes del comando.

Permisos predeterminados

Probemos un experimento. Abra una ventana de terminal y cree un archivo vacío con el siguiente comando:

```
$ touch testfile
```

Ahora, echemos un vistazo a los permisos para este archivo. *Pueden* ser diferentes en su sistema, pero supongamos que tienen el siguiente aspecto:

```
$ ls -lh testfile
-rw-r--r-- 1 carol carol 0 jul 13 21:55 testfile
```

Los permisos son `rw- r- r-`: *read* y *write* para el usuario, y *read* para el grupo y otros, o `644` en modo octal. Ahora, intente crear un directorio:

```
$ mkdir testdir
$ ls -lhd testdir
drwxr-xr-x 2 carol carol 4,0K jul 13 22:01 testdir
```

Ahora los permisos son `rwxr-xr-x`: *read*, *write* y *execute* para el usuario, *read* y *execute* para el grupo y otros, o `755` en modo octal.

No importa dónde se encuentre en el sistema de archivos, cada archivo o directorio que cree obtendrá los mismos permisos. ¿Se ha preguntado alguna vez de dónde vienen?

Vienen de *user mask* o `umask`, que establece los permisos predeterminados para cada archivo creado. Puede comprobar los valores actuales con el comando `umask`:

```
$ umask
0022
```

Pero eso no se parece a `rw- r-- r--`, ni siquiera a `644`. Quizás deberíamos probar con el parámetro `-S`, para obtener una salida en modo simbólico:

```
$ umask -S
u=rwx,g=rx,o=rx
```

Esos son los mismos permisos que obtuvo nuestro directorio de prueba en uno de los ejemplos anteriores. Pero, ¿por qué cuando creamos un archivo los permisos eran diferentes?

Bueno, no tiene sentido establecer permisos de ejecución globales para todos en cualquier archivo de forma predeterminada, ¿verdad? Los directorios necesitan permisos de ejecución (de lo contrario, no puede entrar en ellos), pero los archivos no, por lo que no los obtienen. De ahí el `rw- r-- r--`.

Además de mostrar los permisos predeterminados, `umask` también se puede usar para cambiarlos para su sesión de shell actual. Por ejemplo, si usamos el comando:

```
$ umask u=rwx,g=rwx,o=
```

Cada directorio nuevo heredará los permisos `rwxrwx---`, y cada archivo `rw-rw----` (ya que no obtienen permisos de ejecución). Si repite los ejemplos anteriores para crear un `testfile` y un `testdir` y verifica los permisos, debería obtener:

```
$ ls -lhd test*
drwxrwx--- 2 carol carol 4,0K jul 13 22:25 testdir
-rw-rw---- 1 carol carol    0 jul 13 22:25 testfile
```

Y si marca `umask` sin el parámetro `-S` (modo simbólico), obtiene:

```
$ umask
0007
```

El resultado no parece familiar porque los valores utilizados son diferentes. Aquí hay una tabla con cada valor y su respectivo significado:

Valor	Permiso para archivos	Permiso para directorios
0	rw-	rwx
1	rw-	rw-

Valor	Permiso para archivos	Permiso para directorios
`2`	`r--`	`r-x`
`3`	`r--`	`r--`
`4`	`-w-`	`-wx`
`5`	`-w-`	`-w-`
`6`	`---`	`--x`
`7`	`---`	`---`

Como puede ver, `007` corresponde a `rwxrwx---`, exactamente como lo solicitamos. El cero inicial se puede ignorar.

Permisos especiales

Además de los permisos de lectura, escritura y ejecución para usuario, grupo y otros, cada archivo puede tener otros tres *permisos especiales* que pueden alterar la forma en que funciona un directorio o cómo se ejecuta un programa. Se pueden especificar en modo simbólico u octal, y son los siguientes:

Bit Adhesivo

El bit adhesivo, también llamado *bandera de eliminación restringida*, tiene el valor octal `1` y en modo simbólico está representado por una `t` dentro de los permisos del otro. Esto se aplica solo a los directorios y no tiene ningún efecto en los archivos normales. En Linux, evita que los usuarios eliminen o cambien el nombre de un archivo en un directorio a menos que sean propietarios de ese archivo o directorio.

Los directorios con el bit adhesivo establecido muestran una `t` reemplazando la `x` en los permisos de *otros* en la salida de `ls -l`:

```
$ ls -ld Sample_Directory/
drwxr-xr-t 2 carol carol 4096 Dec 20 18:46 Sample_Directory/
```

En el modo octal, los permisos especiales se especifican mediante una notación de 4 dígitos, donde el primer dígito representa el permiso especial para actuar. Por ejemplo, para establecer el bit adhesivo (valor `1`) para el directorio `Another_Directory` en modo octal, con permisos `755`, el comando sería:

```
$ chmod 1755 Another_Directory
$ ls -ld Another_Directory
drwxr-xr-t 2 carol carol 4,0K Dec 20 18:46 Another_Directory
```

Establecer GID

Establecer GID, también conocido como SGID o Set Group ID bit, tiene el valor octal `2` y en modo simbólico está representado por una `s` en los permisos de *group*. Esto se puede aplicar a archivos o directorios ejecutables. En archivos, hará que el proceso se ejecute con los privilegios del grupo propietario del archivo. Cuando se aplica a directorios, hará que cada archivo o directorio creado bajo él herede el grupo del directorio principal.

Los archivos y directorios con el bit SGID muestran una `s` que reemplaza la `x` en los permisos del *group* en la salida de `ls -l`:

```
$ ls -l test.sh
-rwxr-sr-x 1 carol root 33 Dec 11 10:36 test.sh
```

Para agregar permisos SGID a un archivo en modo simbólico, el comando sería:

```
$ chmod g+s test.sh
$ ls -l test.sh
-rwxr-sr-x 1 carol root     33 Dec 11 10:36 test.sh
```

El siguiente ejemplo le ayudará a comprender mejor los efectos de SGID en un directorio. Supongamos que tenemos un directorio llamado `Sample_Directory`, propiedad del usuario `carol` y del grupo `users`, con la siguiente estructura de permisos:

```
$ ls -ldh Sample_Directory/
drwxr-xr-x 2 carol users 4,0K Jan 18 17:06 Sample_Directory/
```

Ahora, cambiemos a este directorio y, usando el comando `touch`, creemos un archivo vacío dentro de él. El resultado sería:

```
$ cd Sample_Directory/
$ touch newfile
$ ls -lh newfile
-rw-r--r-- 1 carol carol 0 Jan 18 17:11 newfile
```

Como podemos ver, el archivo es propiedad del usuario `carol` y del grupo `carol`. Pero, si el directorio tuviera el permiso SGID establecido, el resultado sería diferente. Primero, agreguemos el bit SGID al `Sample_Directory` y verifiquemos los resultados:

```
$ sudo chmod g+s Sample_Directory/
$ ls -ldh Sample_Directory/
drwxr-sr-x 2 carol users 4,0K Jan 18 17:17 Sample_Directory/
```

La `s` en los permisos de grupo indica que el bit SGID está establecido. Ahora, cambiaremos a este directorio y, nuevamente, crearemos un archivo vacío con el comando `touch`:

```
$ cd Sample_Directory/
$ touch emptyfile
$ ls -lh emptyfile
 -rw-r--r-- 1 carol users 0 Jan 18 17:20 emptyfile
```

El grupo propietario del archivo es `users`. Esto se debe a que el bit SGID hizo que el archivo heredara el propietario del grupo de su directorio principal, que es `users`.

Establecer UID

SUID, también conocido como Establecer ID de usuario, tiene el valor octal `4` y está representado por una `s` en los permisos de *user* en modo simbólico. Solo se aplica a archivos y no tiene ningún efecto en los directorios. Su comportamiento es similar al bit SGID, pero el proceso se ejecutará con los privilegios del *usuario* propietario del archivo. Los archivos con el bit SUID muestran una `s` que reemplaza la `x` en los permisos del usuario en la salida de `ls -l`:

```
$ ls -ld test.sh
-rwsr-xr-x 1 carol carol 33 Dec 11 10:36 test.sh
```

Puede combinar varios permisos especiales en un parámetro. Entonces, para establecer SGID (valor `2`) y SUID (valor `4`) en modo octal para el script `test.sh` con permisos `755`, debe escribir:

```
$ chmod 6755 test.sh
```

Y el resultado sería:

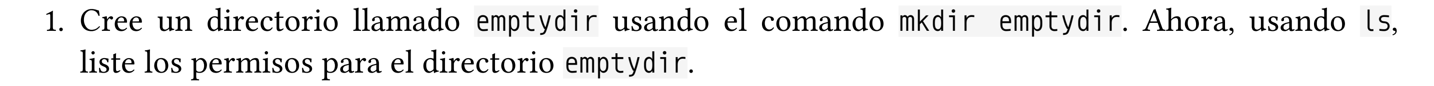
```
$ ls -lh test.sh
-rwsr-sr-x 1 carol carol 66 Jan 18 17:29 test.sh
```

TIP

Si su terminal es compatible con el color, y en estos días la mayoría de ellos lo hacen, puede ver rápidamente si estos permisos especiales se establecen echando un vistazo a la salida de `ls -l`. Para el bit adhesivo, el nombre del directorio podría mostrarse en una fuente negra con fondo azul. Lo mismo se aplica a los archivos con los bits SGID (fondo amarillo) y SUID (fondo rojo). Los colores pueden ser diferentes según la distribución de Linux y la configuración de terminal que utilice.

Ejercicios Guiados

1. Cree un directorio llamado `emptydir` usando el comando `mkdir emptydir`. Ahora, usando `ls`, liste los permisos para el directorio `emptydir`.



2. Cree un archivo vacío llamado `emptyfile` con el comando `touch emptyfile`. Ahora, usando `chmod` en modo simbólico, agregue permisos de ejecución para el propietario del archivo `emptyfile` y elimine los permisos de escritura y ejecución para todos los demás. Haga esto usando solo un comando `chmod`.

3. ¿Cuáles serían los permisos predeterminados para un archivo si el valor de `umask` se establece en `027`?

4. Supongamos que un archivo llamado `test.sh` es un script de shell con los siguientes permisos y propiedad:

```
-rwxr-sr-x 1 carol root     33 Dec 11 10:36 test.sh
```

   - ¿Cuáles son los permisos para el propietario del archivo?

   - Usando la notación octal, ¿cuál sería la sintaxis de `chmod` para “eliminar” el permiso especial otorgado a este archivo?

5. Considere este archivo:

```
$ ls -l /dev/sdb1
brw-rw---- 1 root disk 8, 17 Dec 21 18:51 /dev/sdb1
```

   ¿Qué tipo de archivo es `sdb1`? ¿Quién puede escribir en él?

6. Considere los siguientes 4 archivos:

```
drwxr-xr-t 2 carol carol 4,0K Dec 20 18:46 Another_Directory
----r--r-- 1 carol carol    0 Dec 11 10:55 foo.bar
-rw-rw-r-- 1 carol carol 1,2G Dec 20 18:22 HugeFile.zip
drwxr-sr-x 2 carol users  4,0K Jan 18 17:26 Sample_Directory
```

Anote los permisos correspondientes para cada archivo y directorio usando el modo octal usando la notación de 4 dígitos.

`Another_Directory`	
`foo.bar`	
`HugeFile.zip`	
`Sample_Directory`	

Ejercicios Exploratorios

1. Pruebe esto en una terminal: cree un archivo vacío llamado `emptyfile` con el comando `touch emptyfile`. Ahora "cambie a cero" los permisos para el archivo con `chmod 000 emptyfile`. ¿Qué sucederá si cambia los permisos para `emptyfile` pasando solo *un* valor para `chmod` en modo octal, como `chmod 4 emptyfile`? ¿Y si usa dos, como en `chmod 44 emptyfile`? ¿Qué podemos aprender sobre la forma en que `chmod` lee el valor numérico?

2. Considere los permisos para el directorio temporal en un sistema Linux, `/tmp`:

```
$ ls -l /tmp
drwxrwxrwt  19 root root  16K Dec 21 18:58 tmp
```

   El usuario, el grupo y otros tienen permisos completos. Pero, ¿puede un usuario normal eliminar *cualquier* archivo dentro de este directorio? ¿Por qué es este el caso?

3. Un archivo llamado `test.sh` tiene los siguientes permisos: `-rwsr-xr-x`, lo que significa que el bit SUID está establecido. Ahora, ejecute los siguientes comandos:

```
$ chmod u-x test.sh
$ ls -l test.sh
-rwSr-xr-x 1 carol carol 33 Dec 11 10:36 test.sh
```

   ¿Qué hicimos? ¿Qué significa la `S` mayúscula?

4. ¿Cómo crearía un directorio llamado `Box` donde todos los archivos pertenecen automáticamente al grupo `users` y solo pueden ser eliminados por el usuario que los creó?

Resumen

En esta lección, ha aprendido cómo usar `ls` para obtener (y decodificar) información sobre permisos de archivos, cómo controlar o cambiar quién puede crear, eliminar o modificar un archivo con `chmod`, tanto en los modos *octal* como *simbólico*, cómo cambiar la propiedad de los archivos con `chown` y `chgrp` y cómo consultar y cambiar la máscara de permisos predeterminada para archivos y directorios con `umask`

Los siguientes comandos se discutieron en esta lección:

`ls`
: Lista archivos, opcionalmente incluyendo detalles como permisos.

`chmod`
: Cambia los permisos de un archivo o directorio.

`chown`
: Cambia el usuario y/o grupo propietario de un archivo o directorio.

`chgrp`
: Cambia el grupo propietario de un archivo o directorio.

`umask`
: Consulta o establece la máscara de permisos predeterminada para archivos y directorios

Respuestas a los ejercicios guiados

1. Cree un directorio llamado `emptydir` usando el comando `mkdir emptydir`. Ahora, usando `ls`, liste los permisos para el directorio `emptydir`.

   Agregue el parámetro `-d` a `ls` para ver los atributos de archivo de un directorio, en lugar de enumerar su contenido. Entonces, la respuesta es:

   ```
   ls -l -d emptydir
   ```

   Puntos de bonificación si fusiona los dos parámetros en uno, como en `ls -ld emptydir`.

2. Cree un archivo vacío llamado `emptyfile` con el comando `touch emptyfile`. Ahora, usando `chmod` en modo simbólico, agregue permisos de ejecución para el propietario del archivo `emptyfile` y elimine los permisos de escritura y ejecución para todos los demás. Haga esto usando solo un comando `chmod`.

   Piense en ello de esta manera:

   - “Para el usuario propietario del archivo (`u`) agregue (`+`) los permisos de ejecución (`x`)”, entonces `u+x`.
   - “Para el grupo (`g`) y otros usuarios (`o`), elimine (`-`), los permisos de escritura (`w`) y ejecución (`x`)”, entonces `go-wx`.

     Para combinar estos dos conjuntos de permisos, agregamos una coma entre ellos. Entonces el resultado final es:

     ```
     chmod u+x,go-wx emptyfile
     ```

3. ¿Cuáles serían los permisos predeterminados para un archivo si el valor de `umask` se establece en `027`?

   Los permisos serían `rw-r-----`

4. Supongamos que un archivo llamado `test.sh` es un script de shell con los siguientes permisos y propiedad:

   ```
   -rwxr-sr-x 1 carol root     33 Dec 11 10:36 test.sh
   ```

- ¿Cuáles son los permisos para el propietario del archivo?

  Los permisos para el propietario (2 a 4 caracteres en la salida de `ls -l`) son `rwx`, por lo que la respuesta es: "leer, escribir y ejecutar el archivo".

- Usando la notación octal, ¿cuál debería ser la sintaxis de chmod para "eliminar" el permiso especial otorgado a este archivo?

  Podemos "desactivar" los permisos especiales pasando un cuarto dígito, `0`, a `chmod`. Los permisos actuales son `755`, por lo que el comando debería ser `chmod 0755`.

5. Considere este archivo:

```
$ ls -l /dev/sdb1
brw-rw---- 1 root disk 8, 17 Dec 21 18:51 /dev/sdb1
```

¿Qué tipo de archivo es `sdb1`? ¿Quién puede escribir en él?

El primer carácter en la salida de `ls -l` muestra el tipo de archivo. `b` es un *dispositivo de bloque*, generalmente un disco (interno o externo), conectado a la máquina. El propietario (`root`) y cualquier usuario del grupo `disk` pueden escribir en él.

6. Considere los siguientes 4 archivos:

```
drwxr-xr-t 2 carol carol 4,0K Dec 20 18:46 Another_Directory
----r--r-- 1 carol carol    0 Dec 11 10:55 foo.bar
-rw-rw-r-- 1 carol carol 1,2G Dec 20 18:22 HugeFile.zip
drwxr-sr-x 2 carol users 4,0K Jan 18 17:26 Sample_Directory
```

Anote los permisos correspondientes para cada archivo y directorio usando el modo octal usando la notación de 4 dígitos.

Los permisos correspondientes, en modo octal, son los siguientes:

`Another_Directory`	`1755`. `1` para el bit adhesivo, `755` para los permisos regulares (`rwx` para el usuario, `r-x` para el grupo y otros).
`foo.bar`	`0044`. Sin permisos especiales (por lo que el primer dígito es `0`), sin permisos para el usuario (`---`) y solo lectura (`r-r--`) para grupo y otros.

`HugeFile.zip`	`0664`. Sin permisos especiales, por lo que el primer dígito es `0`. `6` (`rw-`) para el usuario y el grupo, `4` (`r--`) para los demás.
`Sample_Directory`	`2755`. `2` para el bit SGID, `7` (`rwx`) para el usuario, `5` (`r-x`) para el grupo y otros.

Respuestas a ejercicios exploratorios

1. Pruebe esto en una terminal: cree un archivo vacío llamado `emptyfile` con el comando `touch emptyfile`. Ahora “cambie a cero” los permisos para el archivo con `chmod 000 emptyfile`. ¿Qué sucederá si cambia los permisos para `emptyfile` pasando solo *un* valor para `chmod` en modo octal, como `chmod 4 emptyfile`? ¿Y si usa dos, como en `chmod 44 emptyfile`? ¿Qué podemos aprender sobre la forma en que `chmod` lee el valor numérico?

   Recuerde que “cambiamos a cero” los permisos para `emptyfile`. Entonces, su estado inicial sería:

   ```
   ---------- 1 carol carol    0 Dec 11 10:55 emptyfile
   ```

   Ahora, intentemos el primer comando, `chmod 4 emptyfile`:

   ```
   $ chmod 4 emptyfile
   $ ls -l emptyfile
   -------r-- 1 carol carol 0 Dec 11 10:55 emptyfile
   ```

   ¿Ve? Se cambiaron los permisos para *otros*. ¿Y si probamos con dos dígitos, como en `chmod 44 emptyfile`?

   ```
   $ chmod 44 emptyfile
   $ ls -l emptyfile
   ----r--r-- 1 carol carol 0 Dec 11 10:55 emptyfile
   ```

   Ahora, los permisos para *group* y *others* se vieron afectados. De esto, podemos concluir que en modo octal `chmod` lee el valor “al revés”, desde el dígito menos significativo (*otros*) al más significativo (*usuario*). Si pasa un dígito, modifica los permisos de *otros*. Con dos dígitos modifica *grupo* y *otros*, y con tres modifica *usuario*, *grupo* y *otros* y con cuatro dígitos modifica *usuario*, *grupo*, *otros* y los permisos especiales.

2. Considere los permisos para el directorio temporal en un sistema Linux, `/tmp`:

   ```
   $ ls -l /tmp
   drwxrwxrwt  19 root root  16K Dec 21 18:58 tmp
   ```

   El usuario, el grupo y otros tienen permisos completos. Pero, ¿puede un usuario normal eliminar *cualquier* archivo dentro de este directorio? ¿Por qué es este el caso?

`/tmp` es lo que llamamos un directorio *world writeable*, lo que significa que cualquier usuario puede escribir en él. Pero no queremos que un usuario juegue con archivos creados por otros, por lo que se establece el *bit adhesivo* (como lo indica la `t` en los permisos de *otros*). Esto significa que un usuario puede eliminar archivos en `/tmp`, pero solo aquellos creados por él mismo.

3. Un archivo llamado `test.sh` tiene los siguientes permisos: `-rwsr-xr-x`, lo que significa que el bit SUID está establecido. Ahora, ejecute los siguientes comandos:

```
$ chmod u-x test.sh
$ ls -l test.sh
-rwSr-xr-x 1 carol carol 33 Dec 11 10:36 test.sh
```

¿Qué hicimos? ¿Qué significa la `S` mayúscula?

Eliminamos los permisos de ejecución para el usuario propietario del archivo. La `s` (o `t`) toma el lugar de la `x` en la salida de `ls -l`, por lo que el sistema necesita una forma de mostrar si el usuario tiene permisos de ejecución o no. Lo hace cambiando el caso del carácter especial.

Una `s` minúscula en el primer grupo de permisos significa que el usuario propietario del archivo tiene permisos de ejecución y que el bit SUID está configurado. Una `S` mayúscula significa que el usuario propietario del archivo carece de (`-`) permisos de ejecución y que el bit SUID está establecido.

Lo mismo puede decirse de SGID, una `s` minúscula en el segundo grupo de permisos significa que el grupo que posee el archivo tiene permisos de ejecución y que el bit SGID está establecido. Una `S` mayúscula significa que el grupo propietario del archivo carece de permisos de ejecución (`-`) y que el bit SGID está establecido.

Esto también es cierto para el bit adhesivo, representado por la `t` en el tercer grupo de permisos. La `t` minúscula significa un conjunto de bits fijos y que otros tienen permisos de ejecución. La `T` mayúscula significa un conjunto de bits fijos y que otros no tienen permisos de ejecución.

4. ¿Cómo crearía un directorio llamado `Box` donde todos los archivos pertenecen automáticamente al grupo `users` y solo pueden ser eliminados por el usuario que los creó?

Este es un proceso de varios pasos. El primer paso es crear el directorio:

```
$ mkdir Box
```

Queremos que cada archivo creado dentro de este directorio sea asignado automáticamente al

grupo `usuarios`. Podemos hacer esto configurando este grupo como el propietario del directorio y luego configurando el bit SGID en él. También debemos asegurarnos de que cualquier miembro del grupo pueda escribir en ese directorio.

Ya que no nos importa cuáles son los otros permisos, y queremos `cambiar` solo los bits especiales, tiene sentido usar el modo simbólico:

```
$ chown :users Box/
$ chmod g+wxs Box/
```

Tenga en cuenta que si su usuario actual no pertenece al grupo `usuarios`, tendrá que usar el comando `sudo` antes de los comandos anteriores para hacer el cambio como root.

Ahora, para la última parte, asegúrese de que solo el usuario que creó un archivo pueda eliminarlo. Esto se hace estableciendo el bit adhesivo (representado por una `t`) en el directorio. Recuerde que está configurado en los permisos para otros (`o`).

```
$ chmod o+t Box/
```

Los permisos en el directorio `Box` deben ser los siguientes:

```
drwxrwsr-t 2 carol users  4,0K Jan 18 19:09 Box
```

Por supuesto, puede especificar SGID y el bit adhesivo usando solo un comando `chmod`:

```
$ chmod g+wxs,o+t Box/
```

Puntos de bonificación si pensaba en eso.



104.6 Crear y cambiar enlaces duros y simbólicos

Referencia al objetivo del LPI

LPIC-1 version 5.0, Exam 101, Objective 104.6

Importancia

2

Áreas de conocimiento clave

- Crear enlaces.
- Identificar enlaces duros y/o simbólicos.
- Copiar versus enlazar archivos.
- Usar enlaces para facilitar las tareas de administración del sistema.

Lista parcial de archivos, términos y utilidades

- `ln`
- `ls`



104.6 Lección 1

Certificación:	LPIC-1
Versión:	5.0
Tema:	104 Dispositivos, sistemas de archivos Linux, estándar de jerarquía del sistema de archivos
Objetivo:	104.6 Crear y cambiar enlaces físicos y simbólicos
Lección:	1 de 1

Introducción

En Linux, algunos archivos reciben un tratamiento especial ya sea por el lugar en el que se almacenan, como los archivos temporales, o la forma en que interactúan con el sistema de archivos, como los enlaces. En esta lección aprenderá qué son los enlaces y cómo administrarlos.

Entender los Enlaces

Como ya se mencionó, en Linux todo se trata como un archivo. Pero hay un tipo especial de archivo, llamado *link*, y hay dos tipos de enlaces en un sistema Linux:

Enlaces simbólicos

También llamados *enlaces suaves*, apuntan a la ruta de otro archivo. Si borra el archivo al que apunta el enlace (llamado *target*), el enlace seguirá existiendo, pero “deja de funcionar”, ya que ahora apunta a “nada”.

Enlaces duros

Piense en un enlace físico como un segundo nombre para el archivo original. No son duplicados, sino que son una entrada adicional en el sistema de archivos que apunta al mismo lugar (inodo) en el disco.

> **TIP** Un *inodo* es una estructura de datos que almacena atributos para un objeto (como un archivo o directorio) en un sistema de archivos. Entre esos atributos están los permisos, la propiedad y en qué bloques del disco se almacenan los datos del objeto. Piense en ello como una entrada en un índice, de ahí el nombre, que proviene de “index node”.

Trabajar con Enlaces duros

Creación de enlaces duros

El comando para crear un enlace físico en Linux es `ln`. La sintaxis básica es:

```
$ ln TARGET LINK_NAME
```

El `TARGET` ya debe existir (este es el archivo al que apuntará el enlace), y si el objetivo no está en el directorio actual, o si desea crear el enlace en otro lugar, *debe* especificar la ruta completa. Por ejemplo, el comando:

```
$ ln target.txt /home/carol/Documents/hardlink
```

creará un archivo llamado `hardlink` en el directorio `/home/carol/Documents/`, vinculado al archivo `target.txt` en el directorio actual.

Si omite el último parámetro (`LINK_NAME`), se creará un vínculo con el mismo nombre que el objetivo en el directorio actual.

Gestión de enlaces duros

Los enlaces duros son entradas en el sistema de archivos que tienen diferentes nombres pero apuntan a los mismos datos en el disco. Todos estos nombres son iguales y pueden usarse para hacer referencia a un archivo. Si cambia el contenido de uno de los nombres, el contenido de todos los demás nombres que apuntan a ese archivo cambia, ya que todos estos nombres apuntan a los mismos datos. Si elimina uno de los nombres, los otros nombres seguirán funcionando.

Esto sucede porque cuando "borras" un archivo, los datos no se borran del disco. El sistema simplemente elimina la entrada en la tabla del sistema de archivos que apunta al inodo correspondiente a los datos en el disco. Pero si tiene una segunda entrada que apunta al mismo inodo, aún puede acceder a los datos. Piense en ello como dos caminos que convergen en el mismo punto. Incluso si bloquea o redirige una de las carreteras, aún puede llegar al destino utilizando la otra.

Puede verificar esto usando el parámetro `-i` de `ls`. Considere los siguientes contenidos de un directorio:

```
$ ls -li
total 224
3806696 -r--r--r-- 2 carol carol 111702 Jun  7 10:13 hardlink
3806696 -r--r--r-- 2 carol carol 111702 Jun  7 10:13 target.txt
```

El número antes de los permisos es el número de inodo. ¿Nota que tanto el archivo `hardlink` como el archivo `target.txt` tienen el mismo número (`3806696`)? Esto se debe a que uno es un vínculo duro del otro.

Pero, ¿cuál es el original y cuál es el enlace? Realmente no se puede decir, ya que para todos los propósitos prácticos son iguales.

Tenga en cuenta que cada enlace fijo que apunta a un archivo aumenta el *conteo de enlaces* del archivo. Este es el número justo después de los permisos en la salida de `ls -l`. De forma predeterminada, cada archivo tiene un recuento de enlaces de `1` (los directorios tienen un recuento de `2`), y cada vínculo físico a él aumenta el recuento en uno. Entonces, esa es la razón del recuento de enlaces de `2` en los archivos de la lista anterior.

A diferencia de los enlaces simbólicos, solo puede crear enlaces físicos a archivos, y tanto el enlace como el destino deben residir en el mismo sistema de archivos.

Mover y eliminar enlaces duros

Dado que los enlaces duros se tratan como archivos normales, se pueden eliminar con `rm` y renombrarlos o moverlos por el sistema de archivos con `mv`. Y dado que un enlace fijo apunta al mismo inodo del objetivo, se puede mover libremente, sin miedo a "romper" el enlace.

Enlaces Simbólicos

Creación de enlaces simbólicos

El comando utilizado para crear un enlace simbólico también es `ln`, pero con el parámetro `-s` agregado. Al igual que:

```
$ ln -s target.txt /home/carol/Documents/softlink
```

Esto creará un archivo llamado `softlink` en el directorio `/home/carol/Documents/`, apuntando al archivo `target.txt` en el directorio actual.

Al igual que con los enlaces físicos, puede omitir el nombre del enlace para crear un enlace con el mismo nombre que el destino en el directorio actual.

Gestión de enlaces simbólicos

Los enlaces simbólicos apuntan a otra ruta en el sistema de archivos. Puede crear enlaces suaves a archivos *y* directorios, incluso en diferentes particiones. Es bastante fácil detectar un enlace simbólico con la salida del comando `ls`:

```
$ ls -lh
total 112K
-rw-r--r-- 1 carol carol 110K Jun  7 10:13 target.txt
lrwxrwxrwx 1 carol carol   12 Jun  7 10:14 softlink -> target.txt
```

En el ejemplo anterior, el primer carácter de los permisos para el archivo `softlink` es `l`, lo que indica un enlace simbólico. Además, justo después del nombre del archivo, verá el nombre del destino al que apunta el enlace, el archivo `target.txt`.

Tenga en cuenta que en los listados de archivos y directorios, los enlaces suaves siempre muestran los permisos `rwx` para el usuario, el grupo y otros, pero en la práctica los permisos de acceso para ellos son los mismos que los del objetivo.

Mover y eliminar enlaces simbólicos

Al igual que los enlaces físicos, los enlaces simbólicos pueden eliminarse usando `rm` y moverse o renombrarse usando `mv`. Sin embargo, se debe tener especial cuidado al crearlos, para evitar "romper" el enlace si se mueve de su ubicación original.

Al crear enlaces simbólicos, debe tener en cuenta que, a menos que se especifique completamente una ruta, la ubicación del objetivo se interpreta como *relativa* a la ubicación del enlace. Esto puede

crear problemas si se mueve el vínculo o el archivo al que apunta.

Esto es más fácil de entender con un ejemplo. Supongamos que tiene un archivo llamado `original.txt` en el directorio actual y desea crear un enlace simbólico llamado `softlink`. Podrías usar:

```
$ ln -s original.txt softlink
```

Y aparentemente todo estaría bien. Comprobemos con `ls`:

```
$ ls -lh
total 112K
-r--r--r-- 1 carol carol 110K Jun  7 10:13 original.txt
lrwxrwxrwx 1 carol carol   12 Jun  7 19:23 softlink -> original.txt
```

Vea cómo se construye el enlace: `softlink` apunta a (→) `original.txt`. Sin embargo, veamos qué sucede si mueve el enlace al directorio anterior e intentas mostrar su contenido usando el comando `less`:

```
$ mv softlink ../
$ less ../softlink
../softlink: No such file or directory
```

Dado que no se especificó la ruta a `original.txt`, el sistema asume que está en el mismo directorio que el enlace. Cuando esto ya no es cierto, el enlace deja de funcionar.

La forma de evitar esto es especificar siempre la ruta completa al destino al crear el enlace:

```
$ ln -s /home/carol/Documents/original.txt softlink
```

De esta manera, no importa dónde mueva el enlace, seguirá funcionando, porque apunta a la ubicación absoluta del objetivo. Verifique con `ls`:

```
$ ls -lh
total 112K
lrwxrwxrwx 1 carol carol   40 Jun  7 19:34 softlink ->
/home/carol/Documents/original.txt
```

Ejercicios Guiados

1. ¿Cuál es el parámetro para `chmod` en el modo *symbolic* para habilitar el bit sticky en un directorio?

2. Imagina que hay un archivo llamado `document.txt` en el directorio `/home/carol/Documents`. ¿Cuál es el comando para crear un enlace simbólico llamado `text.txt` en el directorio actual?

3. Explique la diferencia entre un vínculo físico a un archivo y una copia de este archivo.

Ejercicios Exploratorios

1. Imagine que dentro de un directorio crea un archivo llamado `recipes.txt`. Dentro de este directorio, también creará un enlace físico a este archivo, llamado `receitas.txt`, y un enlace simbólico (o *soft*) a este llamado `rezepte.txt`.

   ```
   $ touch recipes.txt
   $ ln recipes.txt receitas.txt
   $ ln -s recipes.txt rezepte.txt
   ```

   El contenido del directorio debería ser así:

   ```
   $ ls -lhi
   total 160K
   5388833 -rw-r--r-- 4 carol carol 77K jun 17 17:25 receitas.txt
   5388833 -rw-r--r-- 4 carol carol  0K jun 17 17:25 recipes.txt
   5388837 lrwxrwxrwx 1 carol carol  12 jun 17 17:25 rezepte.txt -> receitas.txt
   ```

   Recuerde que, como enlace físico, `receitas.txt` apunta al mismo inodo que se asigna a `recipes.txt`. ¿Qué pasaría con el enlace suave `rezepte.txt` si se elimina el archivo `receitas.txt`? ¿Por qué?

2. Imagine que tiene una unidad flash conectada a su sistema y montada en `/media/youruser/FlashA`. Desea crear un enlace llamado `schematics.pdf` en su directorio de inicio, apuntando al archivo `esquema.pdf` en la raíz de la unidad flash. Entonces, escribe el comando:

   ```
   $ ln /media/youruser/FlashA/esquema.pdf ~/schematics.pdf
   ```

   ¿Qué pasaría? ¿Por qué?

3. Considere la siguiente salida de `ls -lah`:

```
$ ls -lah
total 3,1M
drwxr-xr-x 2 carol carol 4,0K jun 17 17:27 .
drwxr-xr-x 5 carol carol 4,0K jun 17 17:29 ..
-rw-rw-r-- 1 carol carol 2,8M jun 17 15:45 compressed.zip
-rw-r--r-- 4 carol carol  77K jun 17 17:25 document.txt
-rw-rw-r-- 1 carol carol 216K jun 17 17:25 image.png
-rw-r--r-- 4 carol carol  77K jun 17 17:25 text.txt
```

- ¿Cuántos enlaces apuntan al archivo `document.txt`?

- ¿Son enlaces suaves o duros?

- ¿Qué parámetro debería pasar a `ls` para ver qué inodo ocupa cada archivo?

4. Imagine que tiene en su directorio `~/Documents` un archivo llamado `clients.txt` que contiene algunos nombres de clientes y un directorio llamado `somedir`. Dentro de este hay un archivo *diferente también* llamado `clients.txt` con diferentes nombres. Para replicar esta estructura, use los siguientes comandos.

```
$ cd ~/Documents
$ echo "John, Michael, Bob" > clients.txt
$ mkdir somedir
$ echo "Bill, Luke, Karl" > somedir/clients.txt
```

Luego cree un enlace dentro de `somedir` llamado `partners.txt` apuntando a este archivo, con los comandos:

```
$ cd somedir/
$ ln -s clients.txt partners.txt
```

Entonces, la estructura del directorio es:

```
Documents
|-- clients.txt
`-- somedir
    |-- clients.txt
    `-- partners.txt -> clients.txt
```

Ahora, mueva `partners.txt` de `somedir` a `~/Documents` y liste su contenido.

```
$ cd ~/Documents/
$ mv somedir/partners.txt .
$ less partners.txt
```

¿Seguirá funcionando el enlace? Si es así, ¿qué archivo tendrá su contenido en la lista? ¿Por qué?

5. Considere los siguientes archivos:

```
-rw-r--r-- 1 carol carol 19 Jun 24 11:12 clients.txt
lrwxrwxrwx 1 carol carol 11 Jun 24 11:13 partners.txt -> clients.txt
```

¿Cuáles son los permisos de acceso para `partners.txt`? ¿Por qué?

Resumen

En esta lección aprendimos:

- Qué son los enlaces.
- La diferencia entre enlaces *simbólicos* y *duros*.
- Cómo crear enlaces.
- Cómo mover, renombrar o eliminar estos enlaces.

Los siguientes comandos se discutieron en esta lección:

- `ln`: El comando "link". Por sí mismo, este comando crea un vínculo físico. Con la opción `-s` se puede crear un enlace *simbólico* o *suave*. Recuerde que los enlaces físicos solo pueden residir en la misma partición y sistema de archivos, y los enlaces simbólicos pueden atravesar particiones y sistemas de archivos (incluso el almacenamiento conectado a la red).
- El parámetro `-i` de `ls`, que permite ver el número de inodo de un archivo.

Respuestas a los ejercicios guiados

1. ¿Cuál es el parámetro para `chmod` en el modo *symbolic* para habilitar el bit sticky en un directorio?

   El símbolo del bit sticky en modo simbólico es `t`. Como queremos habilitar (agregar) este permiso al directorio, el parámetro debe ser `+t`.

2. Imagina que hay un archivo llamado `document.txt` en el directorio `/home/carol/Documents`. ¿Cuál es el comando para crear un enlace simbólico llamado `text.txt` en el directorio actual?

   `ln -s` es el comando para crear un enlace simbólico. Dado que debe especificar la ruta completa al archivo al que está vinculando, el comando es:

   ```
   $ ln -s /home/carol/Documents/document.txt text.txt
   ```

3. Explique la diferencia entre un vínculo físico a un archivo y una copia de este archivo.

   Un enlace físico es solo otro nombre para un archivo. Aunque parezca un duplicado del archivo original, a todos los efectos, tanto el enlace como el original son iguales, ya que apuntan a los mismos datos en el disco. Los cambios realizados en el contenido del enlace se reflejarán en el original y viceversa. Una copia es una entidad completamente independiente que ocupa un lugar diferente en el disco. Los cambios en la copia no se reflejarán en el original y viceversa.

Respuestas a ejercicios exploratorios

1. Imagine que dentro de un directorio creas un archivo llamado `recipes.txt`. Dentro de este directorio, también creará un enlace físico a este archivo, llamado `receitas.txt`, y un enlace simbólico (o *soft*) a este llamado `rezepte.txt`.

   ```
   $ touch recipes.txt
   $ ln recipes.txt receitas.txt
   $ ln -s receitas.txt rezepte.txt
   ```

   El contenido del directorio debería ser así:

   ```
   $ ls -lhi
   total 160K
   5388833 -rw-r--r-- 4 carol carol 77K jun 17 17:25 receitas.txt
   5388833 -rw-r--r-- 4 carol carol  0K jun 17 17:25 recipes.txt
   5388837 lrwxrwxrwx 1 carol carol  12 jun 17 17:25 rezepte.txt -> receitas.txt
   ```

   Recuerde que, como enlace físico, `receitas.txt` apunta al mismo inodo que se asigna a `recipes.txt`. ¿Qué pasaría con el enlace suave `rezepte.txt` si se elimina el archivo `receitas.txt`? ¿Por qué?

   El enlace suave `rezepte.txt` dejaría de funcionar. Esto se debe a que los enlaces suaves apuntan a nombres, no a inodos, y el nombre `receitas.txt` ya no existe, incluso si los datos todavía están en el disco con el nombre `recipes.txt`.

2. Imagine que tiene una unidad flash conectada a su sistema y montada en `/media/youruser/FlashA`. Desea crear un enlace llamado `schematics.pdf` en su directorio de inicio, apuntando al archivo `esquema.pdf` en la raíz de la unidad flash. Entonces, escribe el comando:

   ```
   $ ln /media/youruser/FlashA/esquema.pdf ~/schematics.pdf
   ```

   ¿Qué pasaría? ¿Por qué?

   El comando fallaría. El mensaje de error sería `Invalid cross-device link`, y aclara la razón: los enlaces físicos no pueden apuntar a un objetivo en una partición o dispositivo diferente. La única forma de crear un enlace como este es usar un enlace *simbólico* o *suave*, agregando el parámetro `-s` a `ln`.

3. Considere la siguiente salida de `ls -lah`:

```
$ ls -lah
total 3,1M
drwxr-xr-x 2 carol carol 4,0K jun 17 17:27 .
drwxr-xr-x 5 carol carol 4,0K jun 17 17:29 ..
-rw-rw-r-- 1 carol carol 2,8M jun 17 15:45 compressed.zip
-rw-r--r-- 4 carol carol  77K jun 17 17:25 document.txt
-rw-rw-r-- 1 carol carol 216K jun 17 17:25 image.png
-rw-r--r-- 4 carol carol  77K jun 17 17:25 text.txt
```

   - ¿Cuántos enlaces apuntan al archivo `document.txt`?

     Cada archivo comienza con un recuento de enlaces de `1`. Dado que el número de enlaces del archivo es `4`, hay tres enlaces que apuntan a ese archivo.

   - ¿Son enlaces suaves o duros?

     Son enlaces duros, ya que los enlaces suaves no aumentan el número de enlaces de un archivo.

   - ¿Qué parámetro debería pasar a `ls` para ver qué inodo ocupa cada archivo?

     El parámetro es `-i`. El inodo se mostrará como la primera columna en la salida de `ls`, como se muestra a continuación:

```
$ ls -lahi
total 3,1M
5388773 drwxr-xr-x 2 carol carol 4,0K jun 17 17:27 .
5245554 drwxr-xr-x 5 carol carol 4,0K jun 17 17:29 ..
5388840 -rw-rw-r-- 1 carol carol 2,8M jun 17 15:45 compressed.zip
5388833 -rw-r--r-- 4 carol carol  77K jun 17 17:25 document.txt
5388837 -rw-rw-r-- 1 carol carol 216K jun 17 17:25 image.png
5388833 -rw-r--r-- 4 carol carol  77K jun 17 17:25 text.txt
```

4. Imagine que tiene en su directorio `~/Documents` un archivo llamado `clients.txt` que contiene algunos nombres de clientes y un directorio llamado `somedir`. Dentro de este hay un archivo *diferente también* llamado `clients.txt` con diferentes nombres. Para replicar esta estructura, use los siguientes comandos.

```
$ cd ~/Documents
$ echo "John, Michael, Bob" > clients.txt
$ mkdir somedir
$ echo "Bill, Luke, Karl" > somedir/clients.txt
```

Luego cree un enlace dentro de `somedir` llamado `partners.txt` apuntando a este archivo, con los comandos:

```
$ cd somedir/
$ ln -s clients.txt partners.txt
```

Entonces, la estructura del directorio es:

```
Documents
|-- clients.txt
`-- somedir
    |-- clients.txt
    `-- partners.txt -> clients.txt
```

Ahora, mueva `partners.txt` de `somedir` a `~/Documents` y liste su contenido.

```
$ cd ~/Documents/
$ mv somedir/partners.txt .
$ less partners.txt
```

¿Seguirá funcionando el enlace? Si es así, ¿qué archivo tendrá su contenido en la lista? ¿Por qué?

Esto es “complicado”, pero el vínculo funcionará, y el archivo que aparece en la lista será el de `~/Documentos`, que contiene los nombres `John`, `Michael`, `Bob`.

Recuerde que, dado que no especificó la ruta completa al destino `clients.txt` al crear el enlace suave `partners.txt`, la ubicación de destino se interpretará como relativa a la ubicación del enlace, que en este caso es el directorio actual.

Cuando el enlace se movió de `~/Documents/somedir` a `~/Documents`, debería dejar de funcionar, ya que el destino ya no estaba en el mismo directorio que el enlace. Sin embargo, da la casualidad de que hay un archivo llamado `clients.txt` en `~/Documents`, por lo que el enlace apuntará a este archivo, en lugar del destino original dentro de `~/somedir`.

Para evitar esto, siempre especifique la ruta completa al objetivo al crear un enlace simbólico.

5. Considere los siguientes archivos:

```
-rw-r--r-- 1 carol carol 19 Jun 24 11:12 clients.txt
lrwxrwxrwx 1 carol carol 11 Jun 24 11:13 partners.txt -> clients.txt
```

¿Cuáles son los permisos de acceso para `partners.txt`? ¿Por qué?

Los permisos de acceso para `partners.txt` son `rw-r-r--`, ya que los enlaces siempre heredan los mismos permisos de acceso que el destino.



104.7 Encontrar archivos de sistema y ubicar archivos en el lugar correspondiente

Referencia al objetivo del LPI

LPIC-1 version 5.0, Exam 101, Objective 104.7

Importancia

2

Áreas de conocimiento clave

- Entender las ubicaciones correctas de los archivos bajo el criterio del FHS.
- Encontrar archivos y comandos en un sistema Linux.
- Conocer la ubicación y finalidad de archivos y directorios importantes tal como se definen por el FHS.

Lista parcial de archivos, términos y utilidades

- `find`
- `locate`
- `updatedb`
- `whereis`
- `which`
- `type`
- `/etc/updatedb.conf`



104.7 Lección 1

Certificación:	LPIC-1
Versión:	5.0
Tema:	104 Dispositivos, sistemas de archivos Linux, estándar de jerarquía del sistema de archivos
Objetivo:	104.7 Buscar archivos del sistema y colocar archivos en la ubicación correcta
Lección:	1 de 1

Introducción

Las distribuciones de Linux vienen en todas las formas y tamaños, pero una cosa que casi todas comparten es que siguen el *Filesystem Hierarchy Standard* (FHS), que define un “diseño estándar” para el sistema de archivos, lo que facilita mucho la interoperación y la administración del sistema. En esta lección, aprenderá más sobre este estándar y cómo encontrar archivos en un sistema Linux.

El estándar de jerarquía del sistema de archivos

El Estándar de jerarquía del sistema de archivos (FHS) es un esfuerzo de la Fundación Linux para estandarizar la estructura y el contenido del directorio en los sistemas Linux. El cumplimiento del estándar no es obligatorio, pero la mayoría de las distribuciones lo siguen.

NOTE

Aquellos interesados en los detalles de la organización del sistema de archivos pueden leer la especificación FHS 3.0, disponible en múltiples formatos en: http://refspecs.linuxfoundation.org/fhs.shtml

Según el estándar, la estructura básica de directorios es la siguiente:

`/`

Este es el directorio raíz, el directorio más alto de la jerarquía. Todos los demás directorios se encuentran dentro de él. Un sistema de archivos a menudo se compara con un "árbol", por lo que este sería el "tronco" al que están conectadas todas las ramas.

`/bin`

Binarios esenciales, disponibles para todos los usuarios.

`/boot`

Archivos necesarios para el proceso de arranque, incluido el disco RAM inicial (initrd) y el propio kernel de Linux.

`/dev`

Archivos de dispositivo. Estos pueden ser dispositivos físicos conectados al sistema (por ejemplo, `/dev/sda` sería el primer disco SCSI o SATA) o dispositivos virtuales proporcionados por el kernel.

`/etc`

Archivos de configuración específicos del host. Los programas pueden crear subdirectorios en `/etc` para almacenar múltiples archivos de configuración si es necesario.

`/home`

Cada usuario del sistema tiene un directorio "home" para almacenar archivos personales y preferencias, y la mayoría de ellos se encuentran en `/home`. Por lo general, el directorio de inicio es el mismo que el nombre de usuario, por lo que el usuario John tendría su directorio en `/home/john`. Las excepciones son el superusuario (root), que tiene un directorio separado (`/root`) y algunos usuarios del sistema.

`/lib`

Se necesitan bibliotecas compartidas para arrancar el sistema operativo y ejecutar los archivos binarios en `/bin` y `/sbin`.

`/media`

Los medios extraíbles montables por el usuario, como unidades flash, lectores de CD y DVD-ROM, disquetes, tarjetas de memoria y discos externos se montan aquí.

`/mnt`
: Punto de montaje para sistemas de archivos montados temporalmente.

`/opt`
: Paquetes de software de aplicación.

`/root`
: Directorio de inicio del superusuario (root).

`/run`
: Datos variables en tiempo de ejecución.

`/sbin`
: Binarios del sistema

`/srv`
: Datos servidos por el sistema. Por ejemplo, las páginas servidas por un servidor web podrían almacenarse en `/srv/www`.

`/tmp`
: Archivos temporales.

`/usr`
: Datos de usuario de solo lectura, incluidos los datos que necesitan algunas aplicaciones y utilidades secundarias.

`/proc`
: Sistema de archivos virtual que contiene datos relacionados con los procesos en ejecución.

`/var`
: Datos variables escritos durante el funcionamiento del sistema, incluida la cola de impresión, datos de registro, buzones de correo, archivos temporales, caché del navegador, etc.

Tenga en cuenta que algunos de esos directorios, como `/etc`, `/usr` y `/var`, contienen una jerarquía completa de subdirectorios debajo de ellos.

Archivos Temporales

Los archivos temporales son archivos que utilizan los programas para almacenar datos que solo se necesitan durante un período breve. Estos pueden ser los datos de procesos en ejecución, registros de

fallos, archivos temporales de un guardado automático, archivos intermediarios utilizados durante una conversión de archivos, archivos de caché, etc.

Ubicación de los Archivos Temporales

La versión 3.0 del *Filesystem Hierarchy Standard* (FHS) define ubicaciones estándares para archivos temporales en sistemas Linux. Cada ubicación tiene un propósito y comportamiento diferente, y se recomienda que los desarrolladores sigan las convenciones establecidas por FHS al escribir datos temporales en el disco.

`/tmp`
: Según la FHS, los programas no deben asumir que los archivos escritos aquí se conservarán entre las invocaciones de un programa. La recomendación es que este directorio se borre (todos los archivos borrados) durante el arranque del sistema, aunque esto no es obligatorio.

`/var/tmp`
: Otra ubicación para archivos temporales, pero esta *no debe borrarse* durante el arranque del sistema. Los archivos almacenados aquí generalmente persistirán entre reinicios.

`/run`
: Este directorio contiene datos variables en tiempo de ejecución que utilizan los procesos en ejecución, como los archivos de identificación de procesos (`.pid`). Los programas que necesitan más de un archivo en tiempo de ejecución pueden crear subdirectorios aquí. Esta ubicación *debe borrarse* durante el arranque del sistema. El propósito de este directorio alguna vez fue servido por `/var/run`, y en algunos sistemas `/var/run` puede ser un enlace simbólico a `/run`.

Tenga en cuenta que no hay nada que impida que un programa cree archivos temporales en otra parte del sistema, pero es una buena práctica respetar las convenciones establecidas por la FHS.

Buscar archivos

Para buscar archivos en un sistema Linux, puede usar el comando `find`. Esta es una herramienta muy poderosa, llena de parámetros que pueden adaptarse a su comportamiento y modificar la salida exactamente a sus necesidades.

Para empezar, `find` necesita dos argumentos: un punto de partida y qué buscar. Por ejemplo, para buscar todos los archivos en el directorio actual (y subdirectorios) cuyos nombres terminan en `.jpg`, puede usar:


```
$ find . -name '*.jpg'
./pixel_3a_seethrough_1.jpg
./Mate3.jpg
./Expert.jpg
./Pentaro.jpg
./Mate1.jpg
./Mate2.jpg
./Sala.jpg
./Hotbit.jpg
```

Esto coincidirá con cualquier archivo cuyos últimos cuatro caracteres del nombre sean `.jpg`, sin importar lo que venga antes, ya que `*` es un comodín para “cualquier cosa”. Sin embargo, vea lo que sucede si se agrega otro `*` al final del patrón:

```
$ find . -name '*.jpg*'
./pixel_3a_seethrough_1.jpg
./Pentaro.jpg.zip
./Mate3.jpg
./Expert.jpg
./Pentaro.jpg
./Mate1.jpg
./Mate2.jpg
./Sala.jpg
./Hotbit.jpg
```

El archivo `Pentaro.jpg.zip` (resaltado arriba) no se incluyó en la lista anterior, porque incluso si contiene `.jpg` en su nombre, no coincidía con el patrón ya que había caracteres adicionales después de él. El nuevo patrón significa “cualquier cosa `.jpg` cualquier cosa”, por lo que coincide.

TIP Tenga en cuenta que el parámetro `-name` distingue entre mayúsculas y minúsculas. Si desea realizar una búsqueda que no distinga entre mayúsculas y minúsculas, utilice `-iname`.

La expresión `* .jpg` debe colocarse entre comillas simples, para evitar que el shell interprete el patrón en sí. Pruebe sin las comillas y vea qué sucede.

De forma predeterminada, `find` comenzará en el punto de partida y descenderá a través de los subdirectorios (y subdirectorios de esos subdirectorios) que se encuentren. Puede restringir este comportamiento con los parámetros `-maxdepth N`, donde `N` es el número máximo de niveles.

Para buscar solo en el directorio actual, usaría `-maxdepth 1`. Suponga que tiene la siguiente estructura de directorios:

```
directory
▯▯▯ clients.txt
▯▯▯ partners.txt -> clients.txt
▯▯▯ somedir
    ▯▯▯ anotherdir
    ▯▯▯ clients.txt
```

Para buscar dentro de `somedir`, necesitaría usar `-maxdepth 2` (el directorio actual +1 nivel hacia abajo). Para buscar dentro de `anotherdir`, se necesitaría `-maxdepth 3` (el directorio actual +2 niveles hacia abajo). El parámetro `-mindepth N` funciona de manera opuesta buscando solo en directorios *al menos* `N` niveles hacia abajo.

El parámetro `-mount` puede usarse para evitar que `find` caiga dentro de los sistemas de archivos montados. También puede restringir la búsqueda a tipos específicos de sistemas de archivos usando el parámetro `-fstype`. Así que `find /mnt -fstype exfat -iname "\ **report**"` solo buscaría dentro de los sistemas de archivos exFAT montados en `/mnt`.

Búsqueda por atributos

Puede utilizar los siguientes parámetros para buscar archivos con atributos específicos, como los que su usuario puede escribir, tienen un conjunto específico de permisos o tienen un tamaño determinado:

`-user USERNAME`

Coincide con los archivos propiedad del usuario `USERNAME`.

`-group GROUPNAME`

Coincide con archivos propiedad del grupo `GROUPNAME`.

`-readable`

Coincide con archivos que son legibles por el usuario actual.

`-writable`

Coincide con archivos en los que el usuario actual puede escribir.

`-executable`

Busca archivos que son ejecutables por el usuario actual. En el caso de directorios, esto coincidirá

con cualquier directorio que el usuario pueda ingresar (permiso `x`).

`-perm NNNN`

Esto coincidirá con cualquier archivo que tenga exactamente el permiso `NNNN`. Por ejemplo, `-perm 0664` coincidirá con cualquier archivo que el usuario y el grupo puedan leer y escribir y que otros puedan leer (o `rw-rw-r--`).

Puede agregar un `-` antes de `NNNN` para buscar archivos que tengan *al menos* el permiso especificado. Por ejemplo, `-perm -644` coincidiría con archivos que tengan al menos permisos `644` (`rw-r-r--`). Esto incluye un archivo con `664` (`rw-rw-r--`) o incluso `775` (`rwxrwx-r-x`).

`-empty`

Coincidirá con archivos y directorios vacíos.

`-size N`

Coincidirá con cualquier archivo de tamaño `N`, donde `N` por defecto es un número de bloques de 512 bytes. Puede agregar sufijos a `N` para otras unidades: `Nc` contará el tamaño en bytes, `Nk` en kibibytes (KiB, múltiplos de 1024 bytes), `NM` en mebibytes (MiB, múltiplos de 1024 * 1024) y `NG` para gibibytes (GiB, múltiplos de 1024 * 1024 * 1024).

Nuevamente, puede agregar los prefijos `+` o `-` (aquí significa *más grande que* ó *más pequeño que*) para buscar tamaños relativos. Por ejemplo, `-size -10M` coincidirá con cualquier archivo de menos de 10 MiB de tamaño.

Por ejemplo, para buscar archivos en su directorio de inicio que contengan el patrón `report` sin distinguir entre mayúsculas y minúsculas en cualquier parte del nombre, tengan permisos `0644`, hayan sido accedidos hace 10 días y cuyo tamaño sea de al menos 1 Mib, podría utilizar

```
$ find ~ -iname "*report*" -perm 0644 -atime 10 -size +1M
```

Búsqueda por tiempo

Además de buscar atributos, también puede realizar búsquedas por tiempo, encontrando archivos a los que se accedió, se les cambiaron los atributos o se modificaron durante un período de tiempo específico. Los parámetros son:

`-amin N`, `-cmin N`, `-mmin N`

Esto coincidirá con los archivos a los que se ha accedido, se han cambiado los atributos o se han modificado (respectivamente) `N` minutos atrás.

`-atime N`, `-ctime N`, `-mtime N`

Esto coincidirá con los archivos a los que se accedió, se cambiaron los atributos o se modificaron `N*24` horas atrás.

Para `-cmin N` y `-ctime N`, cualquier cambio de atributo provocará una coincidencia, incluido un cambio en los permisos, lectura o escritura en el archivo. Esto hace que estos parámetros sean especialmente poderosos, ya que prácticamente cualquier operación que involucre el archivo activará una coincidencia.

El siguiente ejemplo coincidiría con cualquier archivo del directorio actual que se haya modificado hace menos de 24 horas y tenga un tamaño superior a 100 MiB:

```
$ find . -mtime -1 -size +100M
```

Usando `locate` y `updatedb`

`location` y `updatedb` son comandos que pueden usarse para encontrar rápidamente un archivo que coincida con un patrón dado en un sistema Linux. Pero a diferencia de `find`, `locate` no buscará el patrón en el sistema de archivos: en su lugar, lo buscará en una base de datos construida ejecutando el comando `updatedb`. Esto le da resultados muy rápidos, pero pueden ser imprecisos dependiendo de cuándo se actualizó la base de datos por última vez.

La forma más sencilla de usar `locate` es simplemente darle un patrón para buscar. Por ejemplo, para encontrar todas las imágenes JPEG en su sistema, usaría `locate jpg`. La lista de resultados puede ser bastante extensa, pero debería verse así:

```
$ locate jpg
/home/carol/Downloads/Expert.jpg
/home/carol/Downloads/Hotbit.jpg
/home/carol/Downloads/Mate1.jpg
/home/carol/Downloads/Mate2.jpg
/home/carol/Downloads/Mate3.jpg
/home/carol/Downloads/Pentaro.jpg
/home/carol/Downloads/Sala.jpg
/home/carol/Downloads/pixel_3a_seethrough_1.jpg
/home/carol/Downloads/jpg_specs.doc
```

Cuando se le solicite el patrón `jpg`, `locate` mostrará todo lo que contenga este patrón, sin importar lo que venga antes o después. Puede ver un ejemplo de esto en el archivo `jpg_specs.doc` en la lista de arriba: contiene el patrón, pero la extensión no es `jpg`.

TIP Recuerde que con `locate` está haciendo coincidir patrones, no extensiones de archivo.

Por defecto, el patrón distingue entre mayúsculas y minúsculas. Esto significa que los archivos que contienen `.JPG` no se mostrarán ya que el patrón está en minúsculas. Para evitar esto, pase el parámetro `-i` a `locate`. Repitiendo nuestro ejemplo anterior:

```
$ locate -i .jpg
/home/carol/Downloads/Expert.jpg
/home/carol/Downloads/Hotbit.jpg
/home/carol/Downloads/Mate1.jpg
/home/carol/Downloads/Mate1_old.JPG
/home/carol/Downloads/Mate2.jpg
/home/carol/Downloads/Mate3.jpg
/home/carol/Downloads/Pentaro.jpg
/home/carol/Downloads/Sala.jpg
/home/carol/Downloads/pixel_3a_seethrough_1.jpg
```

Observe que el archivo `Mate1_old.JPG`, en negritas arriba, no estaba presente en la lista anterior.

Puede pasar varios patrones para `locate`, simplemente sepárelos con espacios. El siguiente ejemplo haría una búsqueda que no distingue entre mayúsculas y minúsculas para cualquier archivo que coincida con los patrones `zip` y `jpg`:

```
$ locate -i zip jpg
/home/carol/Downloads/Expert.jpg
/home/carol/Downloads/Hotbit.jpg
/home/carol/Downloads/Mate1.jpg
/home/carol/Downloads/Mate1_old.JPG
/home/carol/Downloads/Mate2.jpg
/home/carol/Downloads/Mate3.jpg
/home/carol/Downloads/OPENMSXPIHAT.zip
/home/carol/Downloads/Pentaro.jpg
/home/carol/Downloads/Sala.jpg
/home/carol/Downloads/gbs-control-master.zip
/home/carol/Downloads/lineage-16.0-20190711-MOD-quark.zip
/home/carol/Downloads/pixel_3a_seethrough_1.jpg
/home/carol/Downloads/jpg_specs.doc
```

Cuando utilice varios patrones, puede solicitar la ubicación para mostrar solo los archivos que coincidan con *todos*. Esto se hace con la opción `-A`. El siguiente ejemplo mostraría cualquier archivo que coincida con los patrones `.jpg` *y* los patrones `.zip`:

```
$ locate -A .jpg .zip
/home/carol/Downloads/Pentaro.jpg.zip
```

Si desea contar el número de archivos que coinciden con un patrón dado en lugar de mostrar su ruta completa, puede usar la opción `-c`. Por ejemplo, para contar el número de archivos `.jpg` en un sistema:

```
$ locate -c .jpg
1174
```

Un problema con `locate` es que solo muestra las entradas presentes en la base de datos generada por `updatedb` (ubicada en `/var/lib/mlocate.db`). Si la base de datos está desactualizada, la salida podría mostrar archivos que se han eliminado desde la última vez que se actualizó. Una forma de evitar esto es agregar el parámetro `-e`, que hará que verifique si el archivo todavía existe antes de mostrarlo en la salida.

Por supuesto, esto no resolverá el problema de los archivos creados *después* de que la última actualización de la base de datos no aparezca. Para ello tendrá que actualizar la base de datos con el comando `updatedb`. El tiempo que llevará esto dependerá de la cantidad de archivos de su disco.

Controlar el comportamiento de `updatedb`

El comportamiento de `updatedb` puede ser controlado por el archivo `/etc/updatedb.conf`. Este es un archivo de texto donde cada línea controla una variable. Las líneas en blanco se ignoran y las líneas que comienzan con el caracter `#` se tratan como comentarios.

`PRUNEFS=`

Cualquier tipo de sistema de archivos indicado después de este parámetro no será escaneado por `updatedb`. La lista de tipos debe estar separada por espacios y los tipos en sí no distinguen entre mayúsculas y minúsculas, por lo que `NFS` y `nfs` son lo mismo.

`PRUNENAMES=`

Esta es una lista de nombres de directorios separados por espacios que no deberían ser escaneados por `updatedb`.

`PRUNEPATHS=`

Esta es una lista de nombres de ruta que deben ser ignorados por `updatedb`. Los nombres de las rutas deben estar separados por espacios y especificados de la misma manera que se mostrarían con `updatedb` (por ejemplo, `/var/spool/media`)

`PRUNE_BIND_MOUNTS=`

Esta es una variable simple `sí` o `no`. Si se establece en `yes`, los montajes de enlace (los directorios montados en otro lugar con el comando `mount --bind`) serán ignorados.

Búsqueda de binarios, páginas del manual y código fuente

`which` es un comando muy útil que muestra la ruta completa a un ejecutable. Por ejemplo, si desea ubicar el ejecutable de `bash`, puede usar:

```
$ which bash
/usr/bin/bash
```

Si se agrega la opción `-a`, el comando mostrará todos los nombres de ruta que coincidan con el ejecutable. Observa la diferencia:

```
$ which mkfs.ext3
/usr/sbin/mkfs.ext3

$ which -a mkfs.ext3
/usr/sbin/mkfs.ext3
/sbin/mkfs.ext3
```

TIP Para encontrar qué directorios están en el `PATH` use el comando `echo $PATH`. Esto imprimirá (`echo`) el contenido de la variable `PATH` (`$PATH`) en su terminal.

`type` es un comando similar que mostrará información sobre un binario, incluyendo dónde se encuentra y su tipo. Simplemente use `type` seguido del nombre del comando:

```
$ type locate
locate is /usr/bin/locate
```

El parámetro `-a` funciona de la misma manera que en `which`, mostrando todos los nombres de ruta que coinciden con el ejecutable. Al igual que:

```
$ type -a locate
locate is /usr/bin/locate
locate is /bin/locate
```

Y el parámetro `-t` mostrará el tipo de archivo del comando, que puede ser `alias`, `keyword`, `function`, `builtin` o `file`. Por ejemplo:

```
$ type -t locate
file

$ type -t ll
alias

$ type -t type
type is a built-in shell command
```

El comando `whereis` es más versátil y, además de los binarios, también se puede usar para mostrar la ubicación de las páginas de manual o incluso el código fuente de un programa (si está disponible en su sistema). Simplemente escriba `whereis` seguido del nombre binario:

```
$ whereis locate
locate: /usr/bin/locate /usr/share/man/man1/locate.1.gz
```

Los resultados anteriores incluyen binarios (`/usr/bin/locate`) y páginas de manual comprimidas (`/usr/share/man/man1/locate.1.gz`).

Puede filtrar rápidamente los resultados utilizando modificadores de línea de comandos como `-b`, que los limitará solo a los binarios, `-m`, que los limitará solo a páginas de manual, o `-s`, que los limitará solo al código fuente. Repitiendo el ejemplo anterior, obtendría:

```
$ whereis -b locate
locate: /usr/bin/locate

$ whereis -m locate
locate: /usr/share/man/man1/locate.1.gz
```

Ejercicios Guiados

1. Imagine que un programa necesita crear un archivo temporal de un solo uso que nunca más será necesario después de que se cierre el programa. ¿Cuál sería el directorio correcto para crear este archivo?

2. ¿Cuál es el directorio temporal que *debe* borrarse durante el proceso de arranque?

3. Usando `find`, busque solo en el directorio actual los archivos que el usuario pueda escribir, que hayan sido modificados en los últimos 10 días y tengan más de 4 GiB.

4. Usando `locate`, busque cualquier archivo que contenga tanto los patrones `report` como `updated`, `update` o `updating` en sus nombres.

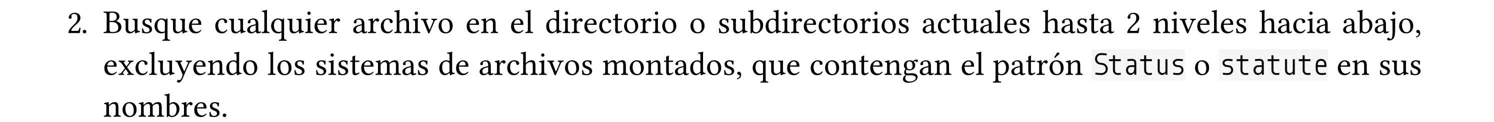
5. ¿Cómo puede encontrar dónde se almacena la página de manual de `ifconfig`?

6. ¿Qué variable debe agregarse a `/etc/updatedb.conf` para que `updatedb` ignore los sistemas de archivos `ntfs`?

7. Un administrador del sistema desea montar un disco interno (`/dev/sdc1`). Según la FHS, ¿en qué directorio se debe montar este disco?

Ejercicios Exploratorios

1. Cuando se usa `locate`, los resultados se extraen de una base de datos generada por `updatedb`. Sin embargo, esta base de datos puede estar desactualizada, lo que hace que `locate` muestre archivos que ya no existen. ¿Cómo se puede hacer que `locate` muestre solo los archivos existentes en su salida?



2. Busque cualquier archivo en el directorio o subdirectorios actuales hasta 2 niveles hacia abajo, excluyendo los sistemas de archivos montados, que contengan el patrón `Status` o `statute` en sus nombres.

3. Limitando la búsqueda a los sistemas de archivos `ext4`, busque cualquier archivo en `/mnt` que tenga al menos permisos de ejecución para el grupo, sea legible para el usuario actual y haya cambiado algún atributo en las últimas 2 horas.

4. Busque archivos vacíos creados hace más de 30 días y que estén al menos dos niveles por debajo del directorio actual

5. Considere que los usuarios `carol` y `john` son parte del grupo `mkt`. Busque en el directorio de inicio de `john` cualquier archivo que también sea legible por `carol`.

Resumen

En esta lección, aprendió acerca de la organización básica del sistema de archivos en una máquina Linux, según el FHS, y cómo buscar binarios y archivos, ya sea por nombre o por atributos. Los siguientes comandos se discutieron en esta lección:

`find`
: Un comando versátil que se utiliza para buscar archivos y carpetas según una variedad de criterios de búsqueda.

`locate`
: Una utilidad que utiliza una base de datos local que contiene las ubicaciones de los archivos almacenados localmente.

`Updatedb`
: Actualiza la base de datos local utilizada por el comando `locate`.

`which`
: Muestra la ruta completa a un ejecutable.

`whereis`
: Muestra las ubicaciones de las páginas del manual, los binarios y el código fuente en el sistema.

`type`
: Muestra la ubicación de un binario y el tipo de aplicación que es (como un programa que está instalado, un programa Bash integrado y más).

Respuestas a los ejercicios guiados

1. Imagine que un programa necesita crear un archivo temporal de un solo uso que nunca más será necesario después de que se cierre el programa. ¿Cuál sería el directorio correcto para crear este archivo?

   Como no nos importa el archivo después de que el programa termine de ejecutarse, el directorio correcto es `/tmp`.

2. ¿Cuál es el directorio temporal que *debe* borrarse durante el proceso de arranque?

   El directorio es `/run` o, en algunos sistemas, `/var/run`.

3. Usando `find`, busque solo en el directorio actual los archivos que el usuario pueda escribir, que hayan sido modificados en los últimos 10 días y tengan un tamaño superior a 4 GiB.

   Para ello, necesitará los parámetros `-writable`, `-mtime` y `-size`:

   ```
   find . -writable -mtime -10 -size +4G
   ```

4. Usando `locate`, busque cualquier archivo que contenga tanto los patrones `report` como `updated`, `update` o `updating` en sus nombres.

   Dado que `locate` debe coincidir con todos los patrones, utilice la opción `-A`:

   ```
   locate -A "report" "updat"
   ```

5. ¿Cómo puede encontrar dónde se almacena la página de manual de `ifconfig`?

   Utilice el parámetro `-m` para `whereis`:

   ```
   whereis -m ifconfig
   ```

6. ¿Qué variable debe agregarse a `/etc/updatedb.conf` para que `updatedb` ignore los sistemas de archivos `ntfs`?

   La variable es `PRUNEFS=` seguida del tipo de sistema de archivos: `PRUNEFS=ntfs`

7. Un administrador del sistema desea montar un disco interno (`/dev/sdc1`). Según la FHS, ¿en qué

directorio se debe montar este disco?

En la práctica, el disco se puede montar en cualquier lugar. Sin embargo, la FHS recomienda que los montajes temporales se realicen en `/mnt`

Respuestas a ejercicios exploratorios

1. Cuando se usa `locate`, los resultados se extraen de una base de datos generada por `updatedb`. Sin embargo, esta base de datos puede estar desactualizada, lo que hace que `locate` muestre archivos que ya no existen. ¿Cómo se puede hacer que `locate` muestre solo los archivos existentes en su salida?

   Agregue el parámetro `-e`, como en `locate -e PATTERN`.

2. Busque cualquier archivo en el directorio o subdirectorios actuales hasta 2 niveles hacia abajo, excluyendo los sistemas de archivos montados, que contengan el patrón `Status` o `statute` en sus nombres.

   Recuerde que para `-maxdepth` también debe considerar el directorio actual, por lo que queremos tres niveles (el actual más 2 niveles hacia abajo):

   ```
   find . -maxdepth 3  -mount -iname "*statu*"
   ```

3. Limitando la búsqueda a los sistemas de archivos `ext4`, busque cualquier archivo en `/mnt` que tenga al menos permisos de ejecución para el grupo, sea legible para el usuario actual y haya cambiado algún atributo en las últimas 2 horas.

   Utilice el parámetro `-fstype` de `mount` para limitar la búsqueda a tipos específicos de sistemas de archivos. Un archivo legible por el usuario actual tendría al menos `4` en el primer dígito de los permisos, y un ejecutable del grupo tendría al menos `1` en el segundo dígito. Como no nos importan los permisos de los demás, podemos usar `0` para el tercer dígito. Use `-cmin N` para filtrar los cambios de atributos recientes, recordando que N se especifica en minutos. Entonces:

   ```
   find /mnt -fstype ext4 -perm -410 -cmin -120
   ```

4. Busque archivos vacíos creados hace más de 30 días y que estén al menos dos niveles por debajo del directorio actual

   El parámetro `-mindepth N` se puede utilizar para limitar la búsqueda al menos a `N` niveles hacia abajo, pero recuerde que debe incluir el directorio actual en la cantidad de niveles. Use `-empty` para verificar archivos vacíos y `-mtime N` para verificar la hora de modificación. Entonces:

   ```
   find . -empty -mtime +30 -mindepth 3
   ```

5. Considere que los usuarios `carol` y `john` son parte del grupo `mkt`. Busque en el directorio de inicio de `john` cualquier archivo que también sea legible por `carol`.

   Teniendo en cuenta que son miembros del mismo grupo, necesitamos al menos una `r` (4) en los permisos del grupo, y no nos importan los demás. Entonces:

   ```
   find /home/john -perm -040
   ```

Pie de imprenta

PDF generado: 2022-04-21

LPI Learning Materials son una iniciativa del Linux Professional Institute (https://lpi.org). Los materiales y sus traducciones pueden encontrarse en https://learning.lpi.org.

Para preguntas y comentarios sobre esta edición, así como sobre todo el proyecto, escriba un correo electrónico a: learning@lpi.org.